

LAMPS Working Group
Internet-Draft
Updates: 4210, 5912, 6712 (if approved)
Intended status: Standards Track
Expires: 31 December 2022

H. Brockhaus, Ed.
D. von Oheimb
Siemens
J. Gray
Entrust
29 June 2022

Certificate Management Protocol (CMP) Updates
draft-ietf-lamps-cmp-updates-23

Abstract

This document contains a set of updates to the syntax and transfer of Certificate Management Protocol (CMP) version 2. This document updates RFC 4210, RFC 5912, and RFC 6712.

The aspects of CMP updated in this document are using EnvelopedData instead of EncryptedValue, clarifying the handling of p10cr messages, improving the crypto agility, as well as adding new general message types, extended key usages to identify certificates for use with CMP, and well-known URI path segments.

CMP version 3 is introduced to enable signaling support of EnvelopedData instead of EncryptedValue and signaling the use of an explicit hash AlgorithmIdentifier in certConf messages, as far as needed.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <https://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 31 December 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction	3
1.1. Convention and Terminology	4
2. Updates to RFC 4210 - Certificate Management Protocol (CMP)	5
2.1. New Section 1.1. - Changes Since RFC 4210	5
2.2. New Section 4.5 - Extended Key Usage	6
2.3. Update Section 5.1.1. - PKI Message Header	7
2.4. New Section 5.1.1.3. - CertProfile	8
2.5. Update Section 5.1.3.1. - Shared Secret Information	9
2.6. Replace Section 5.1.3.4 - Multiple Protection	9
2.7. Replace Section 5.2.2. - Encrypted Values	10
2.8. New Section 5.2.9 - GeneralizedTime	12
2.9. Update Section 5.3.4. - Certification Response	12
2.10. Update Section 5.3.18. - Certificate Confirmation Content	13
2.11. Update Section 5.3.19.2. - Signing Key Pair Types	14
2.12. Update Section 5.3.19.3. - Encryption/Key Agreement Key Pair Types	14
2.13. Replace Section 5.3.19.9. - Revocation Passphrase	15
2.14. New Section 5.3.19.14 - CA Certificates	15
2.15. New Section 5.3.19.15 - Root CA Certificate Update	15
2.16. New Section 5.3.19.16 - Certificate Request Template	16
2.17. New Section 5.3.19.17 - CRL Update Retrieval	18
2.18. Update Section 5.3.21 - Error Message Content	18
2.19. Replace Section 5.3.22 - Polling Request and Response	19
2.20. Update Section 7 - Version Negotiation	24
2.21. Update Section 7.1.1. - Clients Talking to RFC 2510 Servers	25
2.22. Add Section 8.4 - Private Keys for Certificate Signing and CMP Message Protection	25
2.23. Add Section 8.5 - Entropy of Random Numbers, Key Pairs, and Shared Secret Information	25

2.24. Add Section 8.6 - Trust Anchor Provisioning Using CMP Messages	26
2.25. Add Section 8.7 - Authorizing requests for certificates with specific EKUs	27
2.26. Update Appendix B - The Use of Revocation Passphrase . .	27
2.27. Update Appendix C - Request Message Behavioral Clarifications	28
2.28. Update Appendix D.1. - General Rules for Interpretation of These Profiles	29
2.29. Update Appendix D.2. - Algorithm Use Profile	30
2.30. Update Appendix D.4. - Initial Registration/Certification (Basic Authenticated Scheme)	30
3. Updates to RFC 6712 - HTTP Transfer for the Certificate Management Protocol (CMP)	30
3.1. Update Section 1. - Introduction	30
3.2. New Section 1.1. - Changes Since RFC 6712	31
3.3. Replace Section 3.6. - HTTP Request-URI	31
4. IANA Considerations	32
5. Security Considerations	34
6. Acknowledgements	34
7. References	34
7.1. Normative References	34
7.2. Informative References	36
Appendix A. ASN.1 Modules	38
A.1. Update to RFC4210 - 1988 ASN.1 Module	38
A.2. Update to RFC5912 - 2002 ASN.1 Module	52
Appendix B. History of Changes	65
Authors' Addresses	72

1. Introduction

[RFC Editor:

Please perform the following substitution.

* RFCXXXX --> the assigned numerical RFC value for this draft

Please update the following references to associated drafts in progress to reflect their final RFC assignments, if possible:

* I-D.ietf-lamps-cmp-algorithms

* I-D.ietf-lamps-lightweight-cmp-profile

* I-D.ietf-ace-cmpv2-coap-transport

]

While using CMP [RFC4210] in industrial and IoT environments and developing the Lightweight CMP Profile [I-D.ietf-lamps-lightweight-cmp-profile] some limitations were identified in the original CMP specification. This document updates RFC 4210 [RFC4210] and RFC 6712 [RFC6712] to overcome these limitations.

Among others, this document improves the crypto agility of CMP, which means to be flexible to react on future advances in cryptography.

This document also introduces new extended key usages to identify CMP endpoints on registration and certification authorities.

As the main content of RFC 4210 [RFC4210] and RFC 6712 [RFC6712] stays unchanged, this document lists all sections that are updated, replaced, or added to the current text of the respective RFCs.

The authors acknowledge that the style of the document is hard to read because the original RFCs must be read along with this document to get the complete content. The working group decided to use this approach in order to keep the changes to RFC 4210 [RFC4210] and RFC 6712 [RFC6712] to the required minimum. This was meant to speed up the editorial process and to minimize the effort spent on reviewing the whole text of the original documents.

1.1. Convention and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

Technical terminology is used in conformance with RFC 4210 [RFC4210], RFC 4211 [RFC4211], and RFC 5280 [RFC5280]. The following key words are used:

CA: Certification authority, which issues certificates.

RA: Registration authority, an optional system component to which a CA delegates certificate management functions such as authorization checks.

KGA: Key generation authority, which generates key pairs on behalf of an EE. The KGA could be co-located with an RA or a CA.

EE: End entity, a user, device, or service that holds a PKI

certificate. An identifier for the EE is given as its subject of the certificate.

2. Updates to RFC 4210 - Certificate Management Protocol (CMP)

2.1. New Section 1.1. - Changes Since RFC 4210

The following subsection describes feature updates to RFC 4210 [RFC4210]. They are always related to the base specification. Hence, references to the original sections in RFC 4210 [RFC4210] are used whenever possible.

Insert this section at the end of the current Section 1:

1.1. Changes Since RFC 4210

The following updates are made in this document:

- * Add new extended key usages for various CMP server types, e.g., registration authority and certification authority, to express the authorization of the entity identified in the certificate containing the respective extended key usage extension to act as the indicated PKI management entity.
- * Extend the description of multiple protection to cover additional use cases, e.g., batch processing of messages.
- * Offering EnvelopedData as the preferred choice next to EncryptedValue to better support crypto agility in CMP. Note that according to RFC 4211 [RFC4211] section 2.1. point 9 the use of the EncryptedValue structure has been deprecated in favor of the EnvelopedData structure. RFC 4211 [RFC4211] offers the EncryptedKey structure, a choice of EncryptedValue and EnvelopedData for migration to EnvelopedData. For reasons of completeness and consistency the type EncryptedValue has been exchanged in all occurrences in RFC 4210 [RFC4210]. This includes the protection of centrally generated private keys, encryption of certificates, and protection of revocation passphrases. To properly differentiate the support of EnvelopedData instead of EncryptedValue, the CMP version 3 is introduced in case a transaction is supposed to use EnvelopedData.
- * Offering an optional hashAlg field in CertStatus supporting confirmation of certificates signed with signature algorithms, e.g., EdDSA, not directly indicating a specific hash algorithm to use to compute the certHash.

- * Adding new general message types to request CA certificates, a root CA update, a certificate request template, or a CRL update.
- * Extend the usage of polling to p10cr, certConf, rr, genm, and error messages.
- * Delete the mandatory algorithm profile in RFC 4210 Appendix D.2 [RFC4210] and refer to CMP Algorithms Section 7 [I-D.ietf-lamps-cmp-algorithms].

2.2. New Section 4.5 - Extended Key Usage

The following subsection introduces a new extended key usage for CMP servers authorized to centrally generate key pairs on behalf of end entities.

Insert this section at the end of the current Section 4:

4.5. Extended Key Usage

The Extended Key Usage (EKU) extension indicates the purposes for which the certified key pair may be used. It therefore restricts the use of a certificate to specific applications.

A CA may want to delegate parts of its duties to other PKI management entities. This section provides a mechanism to both prove this delegation and enable an automated means for checking the authorization of this delegation. Such delegation may also be expressed by other means, e.g., explicit configuration.

To offer automatic validation for the delegation of a role by a CA to another entity, the certificates used for CMP message protection or signed data for central key generation MUST be issued by the delegating CA and MUST contain the respective EKUs. This proves the authorization of this entity by the delegating CA to act in the given role as described below.

The OIDs to be used for these EKUs are:

```
id-kp-cmcCA OBJECT IDENTIFIER ::= {  
    iso(1) identified-organization(3) dod(6) internet(1)  
    security(5) mechanisms(5) pkix(7) kp(3) 27 }  
  
id-kp-cmcRA OBJECT IDENTIFIER ::= {  
    iso(1) identified-organization(3) dod(6) internet(1)  
    security(5) mechanisms(5) pkix(7) kp(3) 28 }  
  
id-kp-cmKGA OBJECT IDENTIFIER ::= {  
    iso(1) identified-organization(3) dod(6) internet(1)  
    security(5) mechanisms(5) pkix(7) kp(3) 32 }
```

Note: RFC 6402 section 2.10 [RFC6402] specifies OIDs for a CMC CA and a CMC RA. As the functionality of a CA and RA is not specific to using CMC or CMP as the certificate management protocol, these EKUs are re-used by CMP.

The meaning of the id-kp-cmKGA EKU is as follows:

CMP KGA: CMP Key Generation Authorities are CAs or are identified by the id-kp-cmKGA extended key usage. The CMP KGA knows the private key it generated on behalf of the end entity. This is a very sensitive service and needs specific authorization, which by default is with the CA certificate itself. The CA may delegate its authorization by placing the id-kp-cmKGA extended key usage in the certificate used to authenticate the origin of the generated private key. The authorization may also be determined through local configuration of the end entity.

2.3. Update Section 5.1.1. - PKI Message Header

Section 5.1.1 of RFC 4210 [RFC4210] describes the PKI message header. This document introduces the new version 3 indicating support of EnvelopedData as specified in Section 2.7.

Replace the ASN.1 Syntax of PKIHeader and the subsequent description of pvno with the following text:

```

PKIHeader ::= SEQUENCE {
    pvno                INTEGER          { cmp1999(1), cmp2000(2),
                                         cmp2021(3) },
    sender              GeneralName,
    recipient           GeneralName,
    messageTime         [0] GeneralizedTime      OPTIONAL,
    protectionAlg       [1] AlgorithmIdentifier{ALGORITHM, {...}}
                        OPTIONAL,
    senderKID           [2] KeyIdentifier          OPTIONAL,
    recipKID            [3] KeyIdentifier          OPTIONAL,
    transactionID       [4] OCTET STRING          OPTIONAL,
    senderNonce         [5] OCTET STRING          OPTIONAL,
    recipNonce          [6] OCTET STRING          OPTIONAL,
    freeText            [7] PKIFreeText           OPTIONAL,
    generalInfo         [8] SEQUENCE SIZE (1..MAX) OF
                        InfoTypeAndValue          OPTIONAL
}

```

```

PKIFreeText ::= SEQUENCE SIZE (1..MAX) OF UTF8String

```

The usage of pvno values is described in Section 7.

2.4. New Section 5.1.1.3. - CertProfile

Section 5.1.1 of RFC 4210 [RFC4210] defines the PKIHeader and id-it OIDs to be used in the generalInfo field. This section introduces id-it-certProfile.

Insert this section after Section 5.1.1.2:

5.1.1.3. CertProfile

This is used by the EE to indicate specific certificate profiles, e.g., when requesting a new certificate or a certificate request template, see Section 5.3.19.16.

```

id-it-certProfile OBJECT IDENTIFIER ::= {id-it 21}
CertProfileValue ::= SEQUENCE SIZE (1..MAX) OF UTF8String

```

When used in an ir/cr/kur/genm, the value MUST NOT contain more elements than the number of CertReqMsg or InfoTypeAndValue elements and the certificate profile names refer to the elements in the given order.

When used in a p10cr, the value MUST NOT contain multiple certificate profile names.

2.5. Update Section 5.1.3.1. - Shared Secret Information

Section 5.1.3.1 of RFC 4210 [RFC4210] describes the MAC based protection of a PKIMessage using the algorithm id-PasswordBasedMac.

Replace the first paragraph with the following text:

In this case, the sender and recipient share secret information with sufficient entropy (established via out-of-band means or from a previous PKI management operation). PKIProtection will contain a MAC value and the protectionAlg MAY be one of the options described in CMP Algorithms [I-D.ietf-lamps-cmp-algorithms]. The PasswordBasedMac is specified as follows (see also [RFC4211] and [RFC9045]):

Replace the last paragraph with the following text (Note: This fixes Errata ID 2616):

Note: It is RECOMMENDED that the fields of PBMPParameter remain constant throughout the messages of a single transaction (e.g., ir/ip/certConf/pkiConf) to reduce the overhead associated with PasswordBasedMac computation.

2.6. Replace Section 5.1.3.4 - Multiple Protection

Section 5.1.3.4 of RFC 4210 [RFC4210] describes the nested message. This document enables using nested messages also for batch-delivery transport of PKI messages between PKI management entities and with mixed body types.

Replace the text of the section with the following text:

5.1.3.4. Multiple Protection

When receiving a protected PKI message, a PKI management entity such as an RA MAY forward that message adding its own protection (which is a MAC or a signature, depending on the information and certificates shared between the RA and the CA). Additionally, multiple PKI messages MAY be aggregated. There are several use cases for such messages.

- * The RA confirms having validated and authorized a message and forwards the original message unchanged.
- * The RA modifies the message(s) in some way (e.g., adds or modifies particular field values or adds new extensions) before forwarding them, then it MAY create its own desired PKIBody. If the changes made by the RA to PKIMessage break the POP of a certificate request, the RA MUST set the popo field to RAVerified. It MAY

include the original PKIMessage from the EE in the generalInfo field of PKIHeader of a nested message (to accommodate, for example, cases in which the CA wishes to check POP or other information on the original EE message). The infoType to be used in this situation is {id-it 15} (see Section 5.3.19 for the value of id-it) and the infoValue is PKIMessages (contents MUST be in the same order as the message in PKIBody).

- * A PKI management entity collects several messages that are to be forwarded in the same direction and forwards them in a batch. Request messages can be transferred as batch upstream (towards the CA); response or announce messages can be transferred as batch downstream (towards an RA, but not to the EE). This can for instance be used when bridging an off-line connection between two PKI management entities.

These use cases are accomplished by nesting the messages within a new PKI message. The structure used is as follows:

```
NestedMessageContent ::= PKIMessages
```

2.7. Replace Section 5.2.2. - Encrypted Values

Section 5.2.2 of RFC 4210 [RFC4210] describes the use of EncryptedValue to transport encrypted data. This document extends the encryption of data to preferably use EnvelopedData.

Replace the text of the section with the following text:

5.2.2. Encrypted Values

Where encrypted data (in this specification, private keys, certificates, or revocation passphrase) are sent in PKI messages, the EncryptedKey data structure is used.

```
EncryptedKey ::= CHOICE {  
    encryptedValue      EncryptedValue, -- deprecated  
    envelopedData       [0] EnvelopedData }
```

See CRMF [RFC4211] for EncryptedKey and EncryptedValue syntax and CMS [RFC5652] for EnvelopedData syntax. Using the EncryptedKey data structure offers the choice to either use EncryptedValue (for backward compatibility only) or EnvelopedData. The use of the EncryptedValue structure has been deprecated in favor of the EnvelopedData structure. Therefore, it is RECOMMENDED to use EnvelopedData.

Note: The EncryptedKey structure defined in CRMF [RFC4211] is reused here, which makes the update backward compatible. Using the new syntax with the untagged default choice EncryptedValue is bits-on-the-wire compatible with the old syntax.

To indicate support for EnvelopedData the pvno cmp2021 has been introduced. Details on the usage of pvno values is described in Section 7.

The EncryptedKey data structure is used in CMP to transport a private key, certificate, or revocation passphrase in encrypted form.

EnvelopedData is used as follows:

- * It contains only one RecipientInfo structure because the content is encrypted only for one recipient.
- * It may contain a private key in the AsymmetricKeyPackage structure as defined in RFC 5958 [RFC5958] wrapped in a SignedData structure as specified in CMS section 5 [RFC5652] and [RFC8933] signed by the Key Generation Authority.
- * It may contain a certificate or revocation passphrase directly in the encryptedContent field.

The content of the EnvelopedData structure, as specified in CMS section 6 [RFC5652], MUST be encrypted using a newly generated symmetric content-encryption key. This content-encryption key MUST be securely provided to the recipient using one of three key management techniques.

The choice of the key management technique to be used by the sender depends on the credential available at the recipient:

- * Recipient's certificate that contains a key usage extension asserting keyAgreement: The content-encryption key will be protected using the key agreement key management technique, as specified in CMS section 6.2.2 [RFC5652]. This is the preferred technique.
- * Recipient's certificate that contains a key usage extension asserting keyEncipherment: The content-encryption key will be protected using the key transport key management technique, as specified in CMS section 6.2.1 [RFC5652].
- * A password or shared secret: The content-encryption key will be protected using the password-based key management technique, as specified in CMS section 6.2.4 [RFC5652].

2.8. New Section 5.2.9 - GeneralizedTime

The following subsection point implementers to [RFC5280] regarding usage of GeneralizedTime.

Insert this section after Section 5.2.8.4:

5.2.9 GeneralizedTime

GeneralizedTime is a standard ASN.1 type and SHALL be used as specified in RFC 5280 Section 4.1.2.5.2 [RFC5280].

2.9. Update Section 5.3.4. - Certification Response

Section 5.3.4 of RFC 4210 [RFC4210] describes the Certification Response. This document updates the syntax by using the parent structure EncryptedKey instead of EncryptedValue as described in Section 2.7 above. Additionally, it clarifies the certReqId to be used in response to a p10cr message.

Replace the ASN.1 syntax with the following text (Note: This also fixes Errata ID 3949 and 4078):

```

CertRepMessage ::= SEQUENCE {
    caPubs          [1] SEQUENCE SIZE (1..MAX) OF CMPCertificate
                    OPTIONAL,
    response        SEQUENCE OF CertResponse
}

CertResponse ::= SEQUENCE {
    certReqId       INTEGER,
    status          PKIStatusInfo,
    certifiedKeyPair CertifiedKeyPair OPTIONAL,
    rspInfo         OCTET STRING      OPTIONAL
    -- analogous to the id-regInfo-utf8Pairs string defined
    -- for regInfo in CertReqMsg [RFC4211]
}

CertifiedKeyPair ::= SEQUENCE {
    certOrEncCert   CertOrEncCert,
    privateKey      [0] EncryptedKey OPTIONAL,
    -- see [RFC4211] for comment on encoding
    publicationInfo [1] PKIPublicationInfo OPTIONAL
}

CertOrEncCert ::= CHOICE {
    certificate      [0] CMPCertificate,
    encryptedCert    [1] EncryptedKey
}

```

Add the following as a new paragraph right after the ASN.1 syntax:

A pl0cr message contains exactly one CertificationRequestInfo data structure as specified in PKCS#10 [RFC2986] but no certReqId. Therefore, the certReqId in the corresponding certification response (cp) message MUST be set to -1.

Add the following as new paragraphs to the end of the section:

The use of EncryptedKey is described in Section 5.2.2.

Note: To indicate support for EnvelopedData the pvno cmp2021 has been introduced. Details on the usage of different pvno values are described in Section 7.

2.10. Update Section 5.3.18. – Certificate Confirmation Content

This section introduces an optional hashAlg field to the CertStatus type used in certConf messages to explicitly specify the hash algorithm for those certificates where no hash algorithm is specified in the signatureAlgorithm field.

Replace the ASN.1 Syntax of CertStatus with the following text:

```
CertStatus ::= SEQUENCE {  
    certHash      OCTET STRING,  
    certReqId     INTEGER,  
    statusInfo    PKIStatusInfo OPTIONAL,  
    hashAlg [0] AlgorithmIdentifier{DIGEST-ALGORITHM, {...}}  
                OPTIONAL  
}
```

The hashAlg field SHOULD be used only in exceptional cases where the signatureAlgorithm of the certificate to be confirmed does not specify a hash algorithm in the OID or in the parameters. In such cases, e.g., for EdDSA, the hashAlg MUST be used to specify the hash algorithm to be used for calculating the certHash value. Otherwise, the certHash value SHALL be computed using the same hash algorithm as used to create and verify the certificate signature. If hashAlg is used, the CMP version indicated by the certConf message header must be cmp2021(3).

2.11. Update Section 5.3.19.2. - Signing Key Pair Types

The following section clarifies the usage of the Signing Key Pair Types on referencing EC curves.

Insert this note at the end of Section 5.3.19.2:

Note: In case several EC curves are supported, several id-ecPublicKey elements as defined in RFC 5480 [RFC5480] need to be given, one per named curve.

2.12. Update Section 5.3.19.3. - Encryption/Key Agreement Key Pair Types

The following section clarifies the use of the Encryption/Key Agreement Key Pair Types on referencing EC curves.

Insert this note at the end of Section 5.3.19.3:

Note: In case several EC curves are supported, several id-ecPublicKey elements as defined in RFC 5480 [RFC5480] need to be given, one per named curve.

2.13. Replace Section 5.3.19.9. – Revocation Passphrase

Section 5.3.19.9 of RFC 4210 [RFC4210] describes the provisioning of a revocation passphrase for authenticating a later revocation request. This document updates the handling by using the parent structure EncryptedKey instead of EncryptedValue to transport this information as described in Section 2.7 above.

Replace the text of the section with the following text:

5.3.19.9. Revocation Passphrase

This MAY be used by the EE to send a passphrase to a CA/RA for the purpose of authenticating a later revocation request (in the case that the appropriate signing private key is no longer available to authenticate the request). See Appendix B for further details on the use of this mechanism.

GenMsg: {id-it 12}, EncryptedKey
GenRep: {id-it 12}, < absent >

The use of EncryptedKey is described in Section 5.2.2.

2.14. New Section 5.3.19.14 – CA Certificates

The following subsection describes PKI general messages using id-it-caCerts. The intended use is specified in Lightweight CMP Profile Section 4.3 [I-D.ietf-lamps-lightweight-cmp-profile].

Insert this section after Section 5.3.19.13:

2.3.19.14 CA Certificates

This MAY be used by the client to get CA certificates.

GenMsg: {id-it 17}, < absent >
GenRep: {id-it 17}, SEQUENCE SIZE (1..MAX) OF
 CMPCertificate | < absent >

2.15. New Section 5.3.19.15 – Root CA Certificate Update

The following subsection describes PKI general messages using id-it-rootCaCert and id-it-rootCaKeyUpdate. The use is specified in Lightweight CMP Profile Section 4.3 [I-D.ietf-lamps-lightweight-cmp-profile].

Insert this section after new Section 5.3.19.14:

5.3.19.15. Root CA Certificate Update

This MAY be used by the client to get an update of a root CA certificate, which is provided in the body of the request message. In contrast to the ckuann message this approach follows the request/response model.

The EE SHOULD reference its current trust anchor in a TrustAnchor structure in the request body, giving the root CA certificate if available, otherwise the public key value of the trust anchor.

```
GenMsg:    {id-it 20}, RootCaCertValue | < absent >
GenRep:    {id-it 18}, RootCaKeyUpdateContent | < absent >
```

```
RootCaCertValue ::= CMPCertificate
```

```
RootCaKeyUpdateValue ::= RootCaKeyUpdateContent
```

```
RootCaKeyUpdateContent ::= SEQUENCE {
    newWithNew      CMPCertificate,
    newWithOld      [0] CMPCertificate OPTIONAL,
    oldWithNew      [1] CMPCertificate OPTIONAL
}
```

Note: In contrast to CAKeyUpdAnnContent, this type offers omitting newWithOld and oldWithNew in the GenRep message, depending on the needs of the EE.

2.16. New Section 5.3.19.16 - Certificate Request Template

The following subsection introduces the PKI general message using id-it-certReqTemplate. Details are specified in the Lightweight CMP Profile Section 4.3 [I-D.ietf-lamps-lightweight-cmp-profile].

Insert this section after new Section 5.3.19.15:

5.3.19.16. Certificate Request Template

This MAY be used by the client to get a template containing requirements for certificate request attributes and extensions. The controls id-regCtrl-algId and id-regCtrl-rsaKeyLen MAY contain details on the types of subject public keys the CA is willing to certify.

The `id-regCtrl-algId` control MAY be used to identify a cryptographic algorithm, see RFC 5280 Section 4.1.2.7 [RFC5280], other than `rsaEncryption`. The algorithm field SHALL identify a cryptographic algorithm. The contents of the optional parameters field will vary according to the algorithm identified. For example, when the algorithm is set to `id-ecPublicKey`, the parameters identify the elliptic curve to be used, see [RFC5480].

The `id-regCtrl-rsaKeyLen` control SHALL be used for algorithm `rsaEncryption` and SHALL contain the intended modulus bit length of the RSA key.

```
GenMsg:      {id-it 19}, < absent >
GenRep:      {id-it 19}, CertReqTemplateContent | < absent >

CertReqTemplateValue ::= CertReqTemplateContent

CertReqTemplateContent ::= SEQUENCE {
    certTemplate      CertTemplate,
    keySpec           Controls OPTIONAL }

Controls ::= SEQUENCE SIZE (1..MAX) OF AttributeTypeAndValue

id-regCtrl-algId OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) pkix(5) regCtrl(1) 11 }

AlgIdCtrl ::= AlgorithmIdentifier{ALGORITHM, {...}}

id-regCtrl-rsaKeyLen OBJECT IDENTIFIER ::= { iso(1)
    identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) pkix(5) regCtrl(1) 12 }

RsaKeyLenCtrl ::= INTEGER (1..MAX)
```

The `CertReqTemplateValue` contains the prefilled `certTemplate` to be used for a future certificate request. The `publicKey` field in the `certTemplate` MUST NOT be used. In case the PKI management entity wishes to specify supported public-key algorithms, the `keySpec` field MUST be used. One `AttributeTypeAndValue` per supported algorithm or RSA key length MUST be used.

Note: The `Controls` ASN.1 type is defined in CRMF Section 6 [RFC4211]

2.17. New Section 5.3.19.17 - CRL Update Retrieval

The following subsection introduces the PKI general message using `id-it-crlStatusList` and `id-it-crls`. Details are specified in the Lightweight CMP Profile Section 4.3 [I-D.ietf-lamps-lightweight-cmp-profile]. Insert this section after new Section 5.3.19.16:

5.3.19.17. CRL Update Retrieval

This MAY be used by the client to get new CRLs, specifying the source of the CRLs and the `thisUpdate` value of the latest CRL it already has, if available. A CRL source is given either by a `DistributionPointName` or the `GeneralNames` of the issuing CA. The `DistributionPointName` should be treated as an internal pointer to identify a CRL that the server already has and not as a way to ask the server to fetch CRLs from external locations. The server shall provide only those CRLs that are more recent than the ones indicated by the client.

```
GenMsg:    {id-it 22}, SEQUENCE SIZE (1..MAX) OF CRLStatus
GenRep:    {id-it 23}, SEQUENCE SIZE (1..MAX) OF
            CertificateList | < absent >
```

```
CRLSource ::= CHOICE {
    dpn          [0] DistributionPointName,
    issuer       [1] GeneralNames }
```

```
CRLStatus ::= SEQUENCE {
    source        CRLSource,
    thisUpdate    Time OPTIONAL }
```

2.18. Update Section 5.3.21 - Error Message Content

Section 5.3.21 of RFC 4210 [RFC4210] describes the regular use of error messages. This document adds a use by a PKI management entity to initiate delayed delivery in response to `certConf`, `rr`, and `genm` requests and to error messages.

Replace the first sentence of the first paragraph with the following one:

This data structure MAY be used by EE, CA, or RA to convey error info and by a PKI management entity to initiate delayed delivery of responses.

Replace the second paragraph with the following text:

This message MAY be generated at any time during a PKI transaction. If the client sends this request, the server MUST respond with a PKIConfirm response, or another ErrorMsg if any part of the header is not valid. In case a PKI management entity sends an error message to the EE with the pKIStatusInfo field containing the status "waiting", the EE will initiate polling as described in Section 5.3.22. Otherwise, both sides MUST treat this message as the end of the transaction (if a transaction is in progress).

2.19. Replace Section 5.3.22 - Polling Request and Response

Section 5.3.22 of RFC 4210 [RFC4210] describes when and how polling messages are used for ir, cr, and kur messages. This document extends the polling mechanism for outstanding responses to any kind of request message. This update also fixes the inconsistent use of the terms 'rReq' vs. 'pollReq' and 'pRep' vs. 'pollRep'.

Replace Section 5.3.22 with following text:

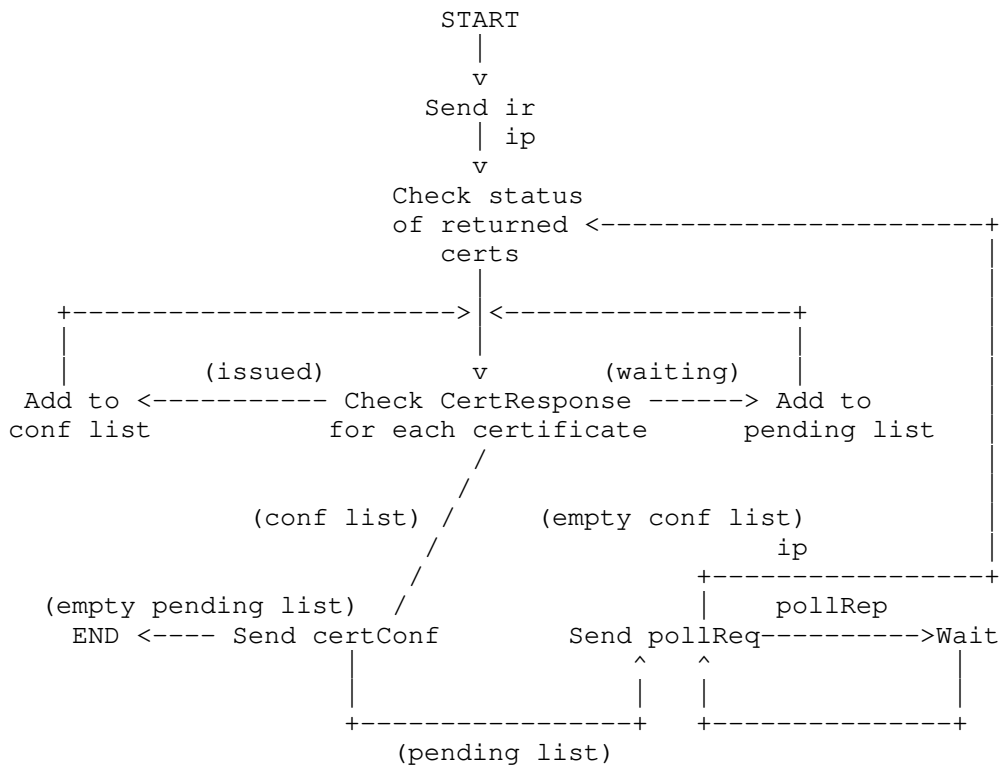
This pair of messages is intended to handle scenarios in which the client needs to poll the server to determine the status of an outstanding response (i.e., when the "waiting" PKIStatus has been received).

```
PollReqContent ::= SEQUENCE OF SEQUENCE {  
    certReqId    INTEGER }  
  
PollRepContent ::= SEQUENCE OF SEQUENCE {  
    certReqId    INTEGER,  
    checkAfter   INTEGER, -- time in seconds  
    reason       PKIFreeText OPTIONAL }
```

In response to an ir, cr, p10cr, or kur request message, polling is initiated with an ip, cp, or kup response message containing status "waiting". For any type of request message, polling can be initiated with an error response messages with status "waiting". The following clauses describe how polling messages are used. It is assumed that multiple certConf messages can be sent during transactions. There will be one sent in response to each ip, cp, or kup that contains a CertStatus for an issued certificate.

- 1 In response to an ip, cp, or kup message, an EE will send a certConf for all issued certificates and expect a PKIConf for each certConf. An EE will send a pollReq message in response to each CertResponse element of an ip, cp, or kup message with status "waiting" and in response to an error message with status "waiting". Its certReqId MUST be either the index of a CertResponse data structure with status "waiting" or -1 referring to the complete response.
- 2 In response to a pollReq, a CA/RA will return an ip, cp, or kup if one or more of still pending requested certificates are ready or the final response to some other type of request is available; otherwise, it will return a pollRep.
- 3 If the EE receives a pollRep, it will wait for at least the number of seconds given in the checkAfter field before sending another pollReq.
- 4 If the EE receives an ip, cp, or kup, then it will be treated in the same way as the initial response; if it receives any other response, then this will be treated as the final response to the original request.

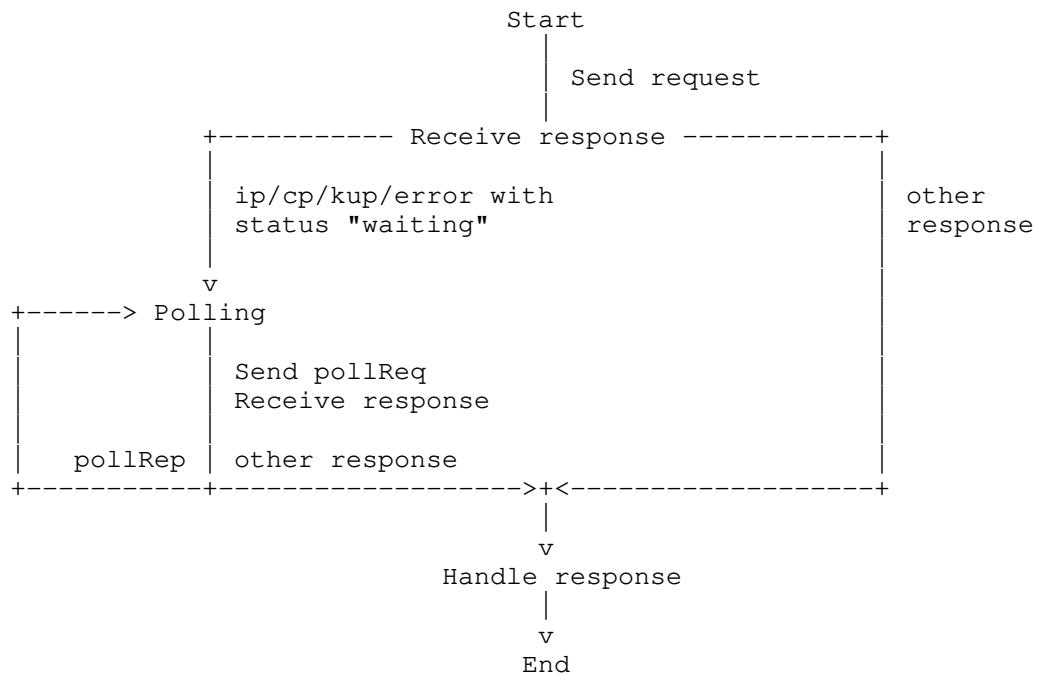
The following client-side state machine describes polling for individual CertResponse elements.



In the following exchange, the end entity is enrolling for two certificates in one request.

Step	End Entity	PKI	
1	Format ir		
2		-> ir	->
3			Handle ir
4			Manual intervention is required for both certs.
5		<- ip	<-
6	Process ip		
7	Format pollReq		
8		-> pollReq	->
9			Check status of cert requests
10			Certificates not ready
11			Format pollRep
12		<- pollRep	<-
13	Wait		
14	Format pollReq		
15		-> pollReq	->
16			Check status of cert requests
17			One certificate is ready
18			Format ip
19		<- ip	<-
20	Handle ip		
21	Format certConf		
22		-> certConf	->
23			Handle certConf
24			Format ack
25		<- pkiConf	<-
26	Format pollReq		
27		-> pollReq	->
28			Check status of certificate
29			Certificate is ready
30			Format ip
31		<- ip	<-
31	Handle ip		
32	Format certConf		
33		-> certConf	->
34			Handle certConf
35			Format ack
36		<- pkiConf	<-

The following client-side state machine describes polling for a complete response message.



In the following exchange, the end-entity is sending a general message request, and the response is delayed by the server.

Step	End Entity	PKI	
1	Format genm		
2		-> genm	->
3			Handle genm
4			delay in response is necessary
5			Format error message "waiting"
			with certReqId set to -1
6		<- error	<-
7	Process error		
8	Format pollReq		
9		-> pollReq	->
10			Check status of original request
			general message response not ready
11			Format pollRep
12		<- pollRep	<-
13	Wait		
14	Format pollReq		
15		-> pollReq	->
16			Check status of original request
			general message response is ready
17			Format genp
18		<- genp	<-
19	Handle genp		

2.20. Update Section 7 - Version Negotiation

Section 7 of RFC 4210 [RFC4210] describes the use of CMP protocol versions. This document describes the handling of the additional CMP version cmp2021 introduced to indicate support of EnvelopedData and hashAlg.

Replace the text of the second paragraph with the following text:

If a client knows the protocol version(s) supported by the server (e.g., from a previous PKIMessage exchange or via some out-of-band means), then it MUST send a PKIMessage with the highest version supported by both it and the server. If a client does not know what version(s) the server supports, then it MUST send a PKIMessage using the highest version it supports, with the following exception. Version cmp2021 SHOULD only be used if cmp2021 syntax is needed for the request being sent or for the expected response.

Note: Using cmp2000 as the default pvno is done to avoid extra message exchanges for version negotiation and to foster compatibility with cmp2000 implementations. Version cmp2021 syntax is only needed if a message exchange uses hashAlg (in CertStatus) or EnvelopedData.

2.21. Update Section 7.1.1. - Clients Talking to RFC 2510 Servers

Section 7.1.1 of RFC 4210 [RFC4210] describes the behavior of a client sending a cmp2000 message talking to a cmp1999 server as specified in RFC 2510 [RFC2510]. This document extends the section to clients with any higher version than cmp1999.

Replace the first sentence of Section 7.1.1 with the following text:

If, after sending a message with a protocol version number higher than cmp1999, a client receives an ErrorMessageContent with a version of cmp1999, then it MUST abort the current transaction.

2.22. Add Section 8.4 - Private Keys for Certificate Signing and CMP Message Protection

The following subsection addresses the risk arising from reusing the CA private key for CMP message protection.

Insert this section after Section 8.3 (Note: This fixes Errata ID 5731):

8.4. Private Keys for Certificate Signing and CMP Message Protection

A CA should not reuse its certificate signing key for other purposes such as protecting CMP responses and TLS connections. This way, exposure to other parts of the system and the number of uses of this particularly critical key is reduced to a minimum.

2.23. Add Section 8.5 - Entropy of Random Numbers, Key Pairs, and Shared Secret Information

The following subsection addresses the risk arising from low entropy of random numbers, asymmetric keys, and shared secret information.

Insert this section after Section 8.4:

8.5. Entropy of Random Numbers, Key Pairs, and Shared Secret Information

Implementations must generate nonces and private keys from random input. The use of inadequate pseudo-random number generators (PRNGs) to generate cryptographic keys can result in little or no security. An attacker may find it much easier to reproduce the PRNG environment that produced the keys and to search the resulting small set of possibilities than brute-force searching the whole key space. As an example of predictable random numbers see [CVE-2008-0166]; consequences of low-entropy random numbers are discussed in Mining

Your Ps and Qs [MiningPsQs]. The generation of quality random numbers is difficult. ISO/IEC 20543:2019 [ISO.20543-2019], NIST SP 800-90A Rev.1 [NIST.SP.800-90Ar1], BSI AIS 31 V2.0 [AIS31], and others offer valuable guidance in this area.

If shared secret information is generated by a cryptographically secure random-number generator (CSRNG) it is safe to assume that the entropy of the shared secret information equals its bit length. If no CSRNG is used, the entropy of a shared secret information depends on the details of the generation process and cannot be measured securely after it has been generated. If user-generated passwords are used as shared secret information, their entropy cannot be measured and are typically insufficient for protected delivery of centrally generated keys or trust anchors.

If the entropy of a shared secret information protecting the delivery of a centrally generated key pair is known, it should not be less than the security strength of that key pair; if the shared secret information is re-used for different key pairs, the security of the shared secret information should exceed the security strength of each individual key pair.

For the case of a PKI management operation that delivers a new trust anchor (e.g., a root CA certificate) using caPubs or genm (a) that is not concluded in a timely manner or (b) where the shared secret information is re-used for several key management operations, the entropy of the shared secret information, if known, should not be less than the security strength of the trust anchor being managed by the operation. The shared secret information should have an entropy that at least matches the security strength of the key material being managed by the operation. Certain use cases may require shared secret information that may be of a low security strength, e.g., a human generated password. It is RECOMMENDED that such secret information be limited to a single PKI management operation.

2.24. Add Section 8.6 - Trust Anchor Provisioning Using CMP Messages

The following subsection addresses the risk arising from in-band provisioning of new trust anchors in a PKI management operation.

Insert this section after new Section 8.5:

8.6. Trust Anchor Provisioning Using CMP Messages

A provider of trust anchors, which may be an RA involved in configuration management of its clients, MUST NOT include to-be-trusted CA certificates in a CMP message unless the specific deployment scenario can ensure that it is adequate that the receiving EE trusts these certificates, e.g., by loading them into its trust store.

Whenever an EE receives in a CMP message, e.g., in the caPubs field of a certificate response or in a general response (genp), a CA certificate for use as a trust anchor, it MUST properly authenticate the message sender with existing trust anchors without requiring new trust anchors included in the message.

Additionally, the EE MUST verify that the sender is an authorized source of trust anchors. This authorization is governed by local policy and typically indicated using shared secret information or with a signature-based message protection using a certificate issued by a PKI that is explicitly authorized for this purpose.

2.25. Add Section 8.7 - Authorizing requests for certificates with specific EKUs

The following subsection addresses the security considerations to follow when authorizing requests for certificates containing specific EKUs.

Insert this section after new Section 8.6:

8.7. Authorizing requests for certificates with specific EKUs

When a CA issues a certificate containing extended key usage extensions as defined in Section 4.5, this expresses delegation of an authorization that originally is only with the CA certificate itself. Such delegation is a very sensitive action in a PKI and therefore special care must be taken when approving such certificate requests to ensure that only legitimate entities receive a certificate containing such an EKU.

2.26. Update Appendix B - The Use of Revocation Passphrase

Appendix B of RFC 4210 [RFC4210] describes the use of the revocation passphrase. As this document updates RFC 4210 [RFC4210] to utilize the parent structure EncryptedKey instead of EncryptedValue as described in Section 2.7 above, the description is updated accordingly.

Replace the first bullet point of this section with the following text:

- * The OID and value specified in Section 5.3.19.9 MAY be sent in a GenMsg message at any time, or MAY be sent in the generalInfo field of the PKIHeader of any PKIMessage at any time. (In particular, the EncryptedKey structure as described in Section 5.2.2 may be sent in the header of the certConf message that confirms acceptance of certificates requested in an initialization request or certificate request message.) This conveys a revocation passphrase chosen by the entity to the relevant CA/RA. When EnvelopedData is used, this is in the decrypted bytes of encryptedContent field. When EncryptedValue is used, this is in the decrypted bytes of the encValue field. Furthermore, the transfer is accomplished with appropriate confidentiality characteristics.

Replace the third bullet point of this section with the following text:

- * Either the localKeyId attribute of EnvelopedData as specified in RFC 2985 [RFC2985] or the valueHint field of EncryptedValue MAY contain a key identifier (chosen by the entity, along with the passphrase itself) to assist in later retrieval of the correct passphrase (e.g., when the revocation request is constructed by the entity and received by the CA/RA).

2.27. Update Appendix C - Request Message Behavioral Clarifications

Appendix C of RFC 4210 [RFC4210] provides clarifications to the request message behavior. As this document updates RFC 4210 [RFC4210] to utilize the parent structure EncryptedKey instead of EncryptedValue as described in Section 2.7 above, the description is updated accordingly.

Replace the comment within the ASN.1 syntax coming after the definition of POPOSigningKey with the following text (Note: This fixes Errata ID 2615):

```

-- *****
-- * For the purposes of this specification, the ASN.1 comment
-- * given in [RFC4211] pertains not only to certTemplate, but
-- * also to the altCertTemplate control.
-- *****
-- * The signature (using "algorithmIdentifier") is on the
-- * DER-encoded value of poposkInput (i.e., the "value" OCTETs
-- * of the POPOSigningKeyInput DER). NOTE: If CertReqMsg
-- * certReq certTemplate (or the altCertTemplate control)
-- * contains the subject and publicKey values, then poposkInput
-- * MUST be omitted and the signature MUST be computed on the
-- * DER-encoded value of CertReqMsg certReq (or the DER-
-- * encoded value of AltCertTemplate). If
-- * certTemplate/altCertTemplate does not contain both the
-- * subject and public key values (i.e., if it contains only
-- * one of these, or neither), then poposkInput MUST be present
-- * and MUST be signed.
-- *****

```

Replace the comment within the ASN.1 syntax coming after the definition of POPOPrivKey with the following text:

```

-- *****
-- * the type of "thisMessage" is given as BIT STRING in RFC 4211
-- * [RFC4211]; it should be "EncryptedKey" (in accordance with
-- * Section 5.2.2 of this specification). Therefore, this
-- * document makes the behavioral clarification of specifying
-- * that the contents of "thisMessage" MUST be encoded either as
-- * "EnvelopedData" or "EncryptedValue" (only for backward
-- * compatibility) and then wrapped in a BIT STRING. This
-- * allows the necessary conveyance and protection of the
-- * private key while maintaining bits-on-the-wire compatibility
-- * with RFC4210 and [RFCXXXX].
-- *****

```

2.28. Update Appendix D.1. - General Rules for Interpretation of These Profiles

Appendix D.1 of RFC 4210 [RFC4210] provides general rules for interpretation of the PKI management messages profiles specified in Appendix D and Appendix E of RFC 4210 [RFC4210]. This document updates a sentence regarding the new protocol version cmp2021.

Replace the last sentence of the first paragraph of the section with the following text:

Mandatory fields are not mentioned if they have an obvious value (e.g., in this version of these profiles, pvno is always cmp2000).

2.29. Update Appendix D.2. - Algorithm Use Profile

Appendix D.2 of RFC 4210 [RFC4210] provides a list of algorithms that implementations must support when claiming conformance with PKI Management Message Profiles as specified in CMP Appendix D.2 [RFC4210]. This document redirects to the new algorithm profile as specified in Section 7.1 of CMP Algorithms [I-D.ietf-lamps-cmp-algorithms].

Replace the text of the section with the following text:

D.2. Algorithm Use Profile

For specifications of algorithm identifiers and respective conventions for conforming implementations, please refer to CMP Algorithms Section 7.1 [I-D.ietf-lamps-cmp-algorithms].

2.30. Update Appendix D.4. - Initial Registration/Certification (Basic Authenticated Scheme)

Appendix D.4 of RFC 4210 [RFC4210] provides the initial registration/certification scheme. This scheme shall continue using EncryptedValue for backward compatibility reasons.

Replace the line specifying protectionAlg of the Initialization Response message with the following text (Note: This fixes Errata ID 5201):

```
protectionAlg      MSG_MAC_ALG
```

Replace the comment after the privateKey field of crc[1].certifiedKeyPair in the syntax of the Initialization Response message with the following text:

```
-- see Appendix C, Request Message Behavioral Clarifications
-- for backward compatibility reasons, use EncryptedValue
```

3. Updates to RFC 6712 - HTTP Transfer for the Certificate Management Protocol (CMP)

3.1. Update Section 1. - Introduction

To indicate and explain why delayed delivery of all kinds of PKIMessages may be handled at transfer level and/or at CMP level, the introduction of RFC 6712 [RFC6712] is updated.

Replace the third paragraph of this section with the following text:

In addition to reliable transport, CMP requires connection and error handling from the transfer protocol, which is all covered by HTTP. Additionally, delayed delivery of CMP response messages may be handled at transfer level regardless of the message contents. Since this document extends the polling mechanism specified in the second version of CMP [RFC4210] to cover all types of PKI management transactions, delays detected at application level may also be handled within CMP, using pollReq and pollRep messages.

3.2. New Section 1.1. - Changes Since RFC 6712

The following subsection describes feature updates to RFC 6712 [RFC6712]. They are related to the base specification. Hence, references to the original sections in RFC 6712 [RFC6712] are used whenever possible.

Insert this section at the end of the current Section 1:

1.1 Changes Since RFC 6712

The following updates are made in this document:

- * Introduce the HTTP path `'/.well-known/cmp'`.
- * Extend the URI structure.

3.3. Replace Section 3.6. - HTTP Request-URI

Section 3.6 of RFC 6712 [RFC6712] specifies the used HTTP URIs. This document introduces the HTTP path `'/.well-known/cmp'` and extends the URIs.

Replace the text of the section with the following text:

3.6. HTTP Request-URI

Each CMP server on a PKI management entity supporting HTTP or HTTPS transfer MUST support the use of the path prefix `'/.well-known/'` as defined in RFC 8615 [RFC8615] and the registered name `'cmp'` to ease interworking in a multi-vendor environment.

The CMP client needs to be configured with sufficient information to form the CMP server URI. This is at least the authority portion of the URI, e.g., `'www.example.com:80'`, or the full operation path segment of the PKI management entity. Additionally, OPTIONAL path segments MAY be added after the registered application name as part of the full operation path to provide further distinction. The path segment `'p'` followed by an arbitraryLabel <name> could for example

support the differentiation of specific CAs or certificate profiles. Further path segments, e.g., as specified in the Lightweight CMP Profile [I-D.ietf-lamps-lightweight-cmp-profile], could indicate PKI management operations using an operationLabel <operation>. A valid full CMP URI can look like this:

```
http://www.example.com/.well-known/cmp
http://www.example.com/.well-known/cmp/<operation>
http://www.example.com/.well-known/cmp/p/<name>
http://www.example.com/.well-known/cmp/p/<name>/<operation>
```

4. IANA Considerations

This document updates the ASN.1 modules of RFC 4210 Appendix F [RFC4210] and RFC 5912 Section 9 [RFC5912]. The OIDs 99 (id-mod-cmp2021-88) and 100 (id-mod-cmp2021-02) were registered in the SMI Security for PKIX Module Identifier registry to identify the updated ASN.1 modules.

This document contains an update to the IANA Consideration sections of [RFC4210] adding this content.

In the SMI-numbers registry "SMI Security for PKIX Extended Key Purpose Identifiers (1.3.6.1.5.5.7.3)" (see <https://www.iana.org/assignments/smi-numbers/smi-numbers.xhtml#smi-numbers-1.3.6.1.5.5.7.3>) as defined in RFC 7299 [RFC7299] one addition has been performed.

One new entry has been added:

Decimal	Description	References
32	id-kp-cmKGA	[RFCXXXX]

Table 1: Addition to the PKIX
Extended Key Purpose Identifiers
Registry

In the SMI-numbers registry "SMI Security for PKIX CMP Information Types (1.3.6.1.5.5.7.4)" (see <https://www.iana.org/assignments/smi-numbers/smi-numbers.xhtml#smi-numbers-1.3.6.1.5.5.7.4>) as defined in RFC 7299 [RFC7299] seven additions have been performed.

Seven new entries have been added:

```
+=====+=====+=====+
```


Decimal	Description	References
17	id-it-caCerts	[RFCXXXX]
18	id-it-rootCaKeyUpdate	[RFCXXXX]
19	id-it-certReqTemplate	[RFCXXXX]
20	id-it-rootCaCert	[RFCXXXX]
21	id-it-certProfile	[RFCXXXX]
22	id-it-crlStatusList	[RFCXXXX]
23	id-it-crls	[RFCXXXX]

Table 2: Addition to the PKIX CMP
Information Types Registry

In the SMI-numbers registry "SMI Security for PKIX CRMF Registration Controls (1.3.6.1.5.5.7.5.1)" (see <https://www.iana.org/assignments/smi-numbers/smi-numbers.xhtml#smi-numbers-1.3.6.1.5.5.7.5.1>) as defined in RFC 7299 [RFC7299] two additions have been performed.

Two new entries have been added:

Decimal	Description	References
11	id-regCtrl-algId	[RFCXXXX]
12	id-regCtrl-rsaKeyLen	[RFCXXXX]

Table 3: Addition to the PKIX CRMF
Registration Controls Registry

This document contains an update to the IANA Consideration sections of [RFC6712] adding this content.

This document defines a new entry with the following content in the "Well-Known URIs" registry (see <https://www.iana.org/assignments/well-known-uris/>) as defined in RFC 8615 [RFC8615].

URI Suffix: cmp
Change Controller: IETF
References: [RFCXXXX] [I-D.ietf-ace-cmpv2-coap-transport]

Related Information: CMP has a sub-registry at
[<https://www.iana.org/assignments/cmp/>]

This document defines a new protocol registry group entitled "Certificate Management Protocol (CMP)" (at <https://www.iana.org/assignments/cmp/>) with a new registry "CMP Well-Known URI Path Segments" containing three columns: Path Segment, Description, and Reference. New items can be added using the Specification Required RFC 8615 [RFC8615] process. The initial contents of this registry is:

Path Segment: p
Description: Indicates that the next path segment specifies, e.g., a CA or certificate profile name
References: [RFCXXXX] [I-D.ietf-ace-cmpv2-coap-transport]

5. Security Considerations

The security considerations of RFC 4210 [RFC4210] are extended in Section 2.22 to Section 2.24. No security considerations updates of RFC 6712 [RFC6712] were required.

6. Acknowledgements

Special thank goes to Jim Schaad for his guidance and the inspiration on structuring and writing this document we got from [RFC6402] which updates CMC. Special thank also goes to Russ Housley, Lijun Liao, Martin Peylo, and Tomas Gustavsson for reviewing and providing valuable suggestions on improving this document.

We also thank all reviewers of this document for their valuable feedback.

7. References

7.1. Normative References

[I-D.ietf-ace-cmpv2-coap-transport]
Sahni, M. and S. Tripathi, "CoAP Transfer for the Certificate Management Protocol", Work in Progress, Internet-Draft, draft-ietf-ace-cmpv2-coap-transport-04, 8 November 2021, <<https://datatracker.ietf.org/doc/html/draft-ietf-ace-cmpv2-coap-transport-04>>.

[I-D.ietf-lamps-cmp-algorithms]
Brockhaus, H., Aschauer, H., Ounsworth, M., and J. Gray, "Certificate Management Protocol (CMP) Algorithms", Work in Progress, Internet-Draft, draft-ietf-lamps-cmp-

algorithms-15, 2 June 2022,
<<https://datatracker.ietf.org/doc/html/draft-ietf-lamps-cmp-algorithms-15>>.

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC2510] Adams, C. and S. Farrell, "Internet X.509 Public Key Infrastructure Certificate Management Protocols", RFC 2510, DOI 10.17487/RFC2510, March 1999, <<https://www.rfc-editor.org/info/rfc2510>>.
- [RFC2985] Nystrom, M. and B. Kaliski, "PKCS #9: Selected Object Classes and Attribute Types Version 2.0", RFC 2985, DOI 10.17487/RFC2985, November 2000, <<https://www.rfc-editor.org/info/rfc2985>>.
- [RFC2986] Nystrom, M. and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7", RFC 2986, DOI 10.17487/RFC2986, November 2000, <<https://www.rfc-editor.org/info/rfc2986>>.
- [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <<https://www.rfc-editor.org/info/rfc3629>>.
- [RFC4210] Adams, C., Farrell, S., Kause, T., and T. Mononen, "Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)", RFC 4210, DOI 10.17487/RFC4210, September 2005, <<https://www.rfc-editor.org/info/rfc4210>>.
- [RFC4211] Schaad, J., "Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)", RFC 4211, DOI 10.17487/RFC4211, September 2005, <<https://www.rfc-editor.org/info/rfc4211>>.
- [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <<https://www.rfc-editor.org/info/rfc5280>>.

- [RFC5480] Turner, S., Brown, D., Yiu, K., Housley, R., and T. Polk, "Elliptic Curve Cryptography Subject Public Key Information", RFC 5480, DOI 10.17487/RFC5480, March 2009, <<https://www.rfc-editor.org/info/rfc5480>>.
- [RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <<https://www.rfc-editor.org/info/rfc5652>>.
- [RFC5958] Turner, S., "Asymmetric Key Packages", RFC 5958, DOI 10.17487/RFC5958, August 2010, <<https://www.rfc-editor.org/info/rfc5958>>.
- [RFC6402] Schaad, J., "Certificate Management over CMS (CMC) Updates", RFC 6402, DOI 10.17487/RFC6402, November 2011, <<https://www.rfc-editor.org/info/rfc6402>>.
- [RFC6712] Kause, T. and M. Peylo, "Internet X.509 Public Key Infrastructure -- HTTP Transfer for the Certificate Management Protocol (CMP)", RFC 6712, DOI 10.17487/RFC6712, September 2012, <<https://www.rfc-editor.org/info/rfc6712>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8615] Nottingham, M., "Well-Known Uniform Resource Identifiers (URIs)", RFC 8615, DOI 10.17487/RFC8615, May 2019, <<https://www.rfc-editor.org/info/rfc8615>>.
- [RFC8933] Housley, R., "Update to the Cryptographic Message Syntax (CMS) for Algorithm Identifier Protection", RFC 8933, DOI 10.17487/RFC8933, October 2020, <<https://www.rfc-editor.org/info/rfc8933>>.
- [RFC9045] Housley, R., "Algorithm Requirements Update to the Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)", RFC 9045, DOI 10.17487/RFC9045, June 2021, <<https://www.rfc-editor.org/info/rfc9045>>.

7.2. Informative References

- [AIS31] Bundesamt fuer Sicherheit in der Informationstechnik (BSI), Killmann, W., and W. Schindler, "A proposal for: Functionality classes for random number generators, version 2.0", 18 September 2011,

https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/Interpretationen/AIS_31_Functionality_classes_for_random_number_generators_e.pdf>.

[CVE-2008-0166]

National Institute of Science and Technology (NIST),
"National Vulnerability Database - CVE-2008-0166", 13 May 2008, <https://nvd.nist.gov/vuln/detail/CVE-2008-0166>>.

[I-D.ietf-lamps-lightweight-cmp-profile]

Brockhaus, H., Oheimb, D. V., and S. Fries, "Lightweight Certificate Management Protocol (CMP) Profile", Work in Progress, Internet-Draft, draft-ietf-lamps-lightweight-cmp-profile-12, 13 May 2022, <https://datatracker.ietf.org/doc/html/draft-ietf-lamps-lightweight-cmp-profile-12>>.

[IEEE.802.1AR_2018]

IEEE, "IEEE Standard for Local and metropolitan area networks - Secure Device Identity", IEEE 802.1AR-2018, DOI 10.1109/IEEESTD.2018.8423794, 2 August 2018, <https://ieeexplore.ieee.org/document/8423794>>.

[ISO.20543-2019]

International Organization for Standardization (ISO),
"Information technology -- Security techniques -- Test and analysis methods for random bit generators within ISO/IEC 19790 and ISO/IEC 15408", ISO Draft Standard 20543-2019, October 2019.

[MiningPsQs]

Security'12: Proceedings of the 21st USENIX conference on Security symposium, Heninger, N., Durumeric, Z., Wustrow, E., and J. A. Halderman, "Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices", August 2012, <https://www.usenix.org/conference/usenixsecurity12/technical-sessions/presentation/heninger>>.

[NIST.SP.800-90Ar1]

Barker, Elaine B. and John M. Kelsey, "Recommendation for Random Number Generation Using Deterministic Random Bit Generators", NIST NIST SP 800-90Ar1, DOI 10.6028/NIST.SP.800-90Ar1, June 2015, <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf>>.

- [PKCS11] RSA Laboratories, "The Public-Key Cryptography Standards - Cryptographic Token Interface Standard. Version 2.10", December 1999,
<<https://www.cryptsoft.com/pkcs11doc/STANDARD/pkcs11v2-10.pdf>>.
- [RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997,
<<https://www.rfc-editor.org/info/rfc2104>>.
- [RFC2202] Cheng, P. and R. Glenn, "Test Cases for HMAC-MD5 and HMAC-SHA-1", RFC 2202, DOI 10.17487/RFC2202, September 1997,
<<https://www.rfc-editor.org/info/rfc2202>>.
- [RFC5912] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)", RFC 5912, DOI 10.17487/RFC5912, June 2010,
<<https://www.rfc-editor.org/info/rfc5912>>.
- [RFC7299] Housley, R., "Object Identifier Registry for the PKIX Working Group", RFC 7299, DOI 10.17487/RFC7299, July 2014,
<<https://www.rfc-editor.org/info/rfc7299>>.

Appendix A. ASN.1 Modules

A.1. Update to RFC4210 - 1988 ASN.1 Module

This section contains the updated ASN.1 module for [RFC4210]. This module replaces the module in Appendix F of that document. Although a 2002 ASN.1 module is provided, this 1988 ASN.1 module remains the normative module as per the policy of the PKIX working group.

```
PKIXCMP {iso(1) identified-organization(3)
  dod(6) internet(1) security(5) mechanisms(5) pkix(7)
  id-mod(0) id-mod-cmp2021-88(99)}

DEFINITIONS EXPLICIT TAGS ::=

BEGIN

-- EXPORTS ALL --

IMPORTS

    Certificate, CertificateList, Extensions, Name, Time,
    AlgorithmIdentifier, id-kp
    --, UTF8String -- -- if required; otherwise, comment out
```

```

        FROM PKIX1Explicit88 {iso(1) identified-organization(3)
        dod(6) internet(1) security(5) mechanisms(5) pkix(7)
        id-mod(0) id-pkix1-explicit-88(18)}
-- The import of Name is added to define CertificationRequest
-- instead of importing it from PKCS#10 [RFC2986]

DistributionPointName, GeneralNames, GeneralName, KeyIdentifier
        FROM PKIX1Implicit88 {iso(1) identified-organization(3)
        dod(6) internet(1) security(5) mechanisms(5) pkix(7)
        id-mod(0) id-pkix1-implicit-88(19)}

CertTemplate, PKIPublicationInfo, EncryptedKey, CertId,
CertReqMessages, Controls, AttributeTypeAndValue, id-regCtrl
        FROM PKIXCRMF-2005 {iso(1) identified-organization(3)
        dod(6) internet(1) security(5) mechanisms(5) pkix(7)
        id-mod(0) id-mod-crmf2005(36)}
-- The import of EncryptedKey is added due to the updates made
-- in CMP Updates [RFCXXXX]. EncryptedValue does not need to
-- be imported anymore and is therefore removed here.

-- see also the behavioral clarifications to CRMF codified in
-- Appendix C of this specification

EnvelopedData, SignedData, Attribute
        FROM CryptographicMessageSyntax2004 { iso(1)
        member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
        smime(16) modules(0) cms-2004(24) }
-- The import of EnvelopedData and SignedData is added due to
-- the updates made in CMP Updates [RFCXXXX]
-- The import of Attribute is added to define
-- CertificationRequest instead of importing it from
-- PKCS#10 [RFC2986]

;

-- the rest of the module contains locally-defined OIDs and
-- constructs

CMPCertificate ::= CHOICE {
    x509v3PKCert      Certificate
}
-- This syntax, while bits-on-the-wire compatible with the
-- standard X.509 definition of "Certificate", allows the
-- possibility of future certificate types (such as X.509
-- attribute certificates, WAP WTLS certificates, or other kinds
-- of certificates) within this certificate management protocol,
-- should a need ever arise to support such generality. Those
-- implementations that do not foresee a need to ever support

```

```

-- other certificate types MAY, if they wish, comment out the
-- above structure and "un-comment" the following one prior to
-- compiling this ASN.1 module. (Note that interoperability
-- with implementations that don't do this will be unaffected by
-- this change.)

-- CMPCertificate ::= Certificate

PKIMessage ::= SEQUENCE {
    header          PKIHeader,
    body            PKIBody,
    protection      [0] PKIProtection OPTIONAL,
    extraCerts      [1] SEQUENCE SIZE (1..MAX) OF CMPCertificate
                    OPTIONAL
}

PKIMessages ::= SEQUENCE SIZE (1..MAX) OF PKIMessage

PKIHeader ::= SEQUENCE {
    pvno            INTEGER          { cmp1999(1), cmp2000(2),
                                     cmp2021(3) },
    sender          GeneralName,
    -- identifies the sender
    recipient       GeneralName,
    -- identifies the intended recipient
    messageTime     [0] GeneralizedTime OPTIONAL,
    -- time of production of this message (used when sender
    -- believes that the transport will be "suitable"; i.e.,
    -- that the time will still be meaningful upon receipt)
    protectionAlg   [1] AlgorithmIdentifier OPTIONAL,
    -- algorithm used for calculation of protection bits
    senderKID       [2] KeyIdentifier OPTIONAL,
    recipKID        [3] KeyIdentifier OPTIONAL,
    -- to identify specific keys used for protection
    transactionID   [4] OCTET STRING OPTIONAL,
    -- identifies the transaction; i.e., this will be the same in
    -- corresponding request, response, certConf, and PKIConf
    -- messages
    senderNonce     [5] OCTET STRING OPTIONAL,
    recipNonce      [6] OCTET STRING OPTIONAL,
    -- nonces used to provide replay protection, senderNonce
    -- is inserted by the creator of this message; recipNonce
    -- is a nonce previously inserted in a related message by
    -- the intended recipient of this message
    freeText        [7] PKIFreeText OPTIONAL,
    -- this may be used to indicate context-specific instructions
    -- (this field is intended for human consumption)
    generalInfo     [8] SEQUENCE SIZE (1..MAX) OF

```



```

                                InfoTypeAndValue    OPTIONAL
    -- this may be used to convey context-specific information
    -- (this field not primarily intended for human consumption)
}

PKIFreeText ::= SEQUENCE SIZE (1..MAX) OF UTF8String
    -- text encoded as UTF-8 String [RFC3629]

PKIBody ::= CHOICE {
    -- message-specific body elements
    ir      [0]  CertReqMessages,      --Initialization Request
    ip      [1]  CertRepMessage,       --Initialization Response
    cr      [2]  CertReqMessages,      --Certification Request
    cp      [3]  CertRepMessage,       --Certification Response
    p10cr   [4]  CertificationRequest, --imported from [RFC2986]
    popdecc [5]  POPODecKeyChallContent, --pop Challenge
    popdecr [6]  POPODecKeyRespContent, --pop Response
    kur     [7]  CertReqMessages,      --Key Update Request
    kup     [8]  CertRepMessage,       --Key Update Response
    krr     [9]  CertReqMessages,      --Key Recovery Request
    krp     [10] KeyRecRepContent,      --Key Recovery Response
    rr      [11] RevReqContent,        --Revocation Request
    rp      [12] RevRepContent,        --Revocation Response
    ccr     [13] CertReqMessages,      --Cross-Cert. Request
    ccp     [14] CertRepMessage,       --Cross-Cert. Response
    ckuann  [15] CAKeyUpdAnnContent,   --CA Key Update Ann.
    cann    [16] CertAnnContent,      --Certificate Ann.
    rann    [17] RevAnnContent,        --Revocation Ann.
    crlann  [18] CRLAnnContent,        --CRL Announcement
    pkiconf [19] PKIConfirmContent,    --Confirmation
    nested  [20] NestedMessageContent, --Nested Message
    genm    [21] GenMsgContent,        --General Message
    genp    [22] GenRepContent,        --General Response
    error   [23] ErrorMsgContent,      --Error Message
    certConf [24] CertConfirmContent,  --Certificate confirm
    pollReq [25] PollReqContent,       --Polling request
    pollRep [26] PollRepContent        --Polling response
}

PKIProtection ::= BIT STRING

ProtectedPart ::= SEQUENCE {
    header    PKIHeader,
    body      PKIBody
}

id-PasswordBasedMac OBJECT IDENTIFIER ::= {1 2 840 113533 7 66 13}
PBMPParameter ::= SEQUENCE {
    salt      OCTET STRING,

```

```
-- note: implementations MAY wish to limit acceptable sizes
-- of this string to values appropriate for their environment
-- in order to reduce the risk of denial-of-service attacks
owf          AlgorithmIdentifier,
-- AlgId for a One-Way Function
iterationCount  INTEGER,
-- number of times the OWF is applied
-- note: implementations MAY wish to limit acceptable sizes
-- of this integer to values appropriate for their environment
-- in order to reduce the risk of denial-of-service attacks
mac          AlgorithmIdentifier
-- the MAC AlgId (e.g., DES-MAC, Triple-DES-MAC [PKCS11],
} -- or HMAC [RFC2104, RFC2202])

id-DHBasedMac OBJECT IDENTIFIER ::= {1 2 840 113533 7 66 30}
DHBMParameter ::= SEQUENCE {
    owf          AlgorithmIdentifier,
    -- AlgId for a One-Way Function
    mac          AlgorithmIdentifier
    -- the MAC AlgId (e.g., DES-MAC, Triple-DES-MAC [PKCS11],
} -- or HMAC [RFC2104, RFC2202])

NestedMessageContent ::= PKIMessages

PKIStatus ::= INTEGER {
    accepted          (0),
    -- you got exactly what you asked for
    grantedWithMods   (1),
    -- you got something like what you asked for; the
    -- requester is responsible for ascertaining the differences
    rejection         (2),
    -- you don't get it, more information elsewhere in the message
    waiting           (3),
    -- the request body part has not yet been processed; expect to
    -- hear more later (note: proper handling of this status
    -- response MAY use the polling req/rep PKIMessages specified
    -- in Section 5.3.22 of [RFC4210]; alternatively, polling in the
    -- underlying transport layer MAY have some utility in this
    -- regard)
    revocationWarning (4),
    -- this message contains a warning that a revocation is
    -- imminent
    revocationNotification (5),
    -- notification that a revocation has occurred
    keyUpdateWarning  (6),
    -- update already done for the oldCertId specified in
    -- CertReqMsg
```

```
}

PKIFailureInfo ::= BIT STRING {
-- since we can fail in more than one way!
-- More codes may be added in the future if/when required.
    badAlg          (0),
    -- unrecognized or unsupported Algorithm Identifier
    badMessageCheck (1),
    -- integrity check failed (e.g., signature did not verify)
    badRequest      (2),
    -- transaction not permitted or supported
    badTime         (3),
    -- messageTime was not sufficiently close to the system time,
    -- as defined by local policy
    badCertId       (4),
    -- no certificate could be found matching the provided criteria
    badDataFormat   (5),
    -- the data submitted has the wrong format
    wrongAuthority  (6),
    -- the authority indicated in the request is different from the
    -- one creating the response token
    incorrectData   (7),
    -- the requester's data is incorrect (for notary services)
    missingTimeStamp (8),
    -- when the timestamp is missing but should be there
    -- (by policy)
    badPOP          (9),
    -- the proof-of-possession failed
    certRevoked     (10),
    -- the certificate has already been revoked
    certConfirmed   (11),
    -- the certificate has already been confirmed
    wrongIntegrity  (12),
    -- not valid integrity, password based instead of signature or
    -- vice versa
    badRecipientNonce (13),
    -- not valid recipient nonce, either missing or wrong value
    timeNotAvailable (14),
    -- the TSA's time source is not available
    unacceptedPolicy (15),
    -- the requested TSA policy is not supported by the TSA.
    unacceptedExtension (16),
    -- the requested extension is not supported by the TSA.
    addInfoNotAvailable (17),
    -- the additional information requested could not be
    -- understood or is not available
    badSenderNonce  (18),
    -- not valid sender nonce, either missing or wrong size
```

```
badCertTemplate      (19),
  -- not valid cert. template or missing mandatory information
signerNotTrusted     (20),
  -- signer of the message unknown or not trusted
transactionIdInUse   (21),
  -- the transaction identifier is already in use
unsupportedVersion    (22),
  -- the version of the message is not supported
notAuthorized        (23),
  -- the sender was not authorized to make the preceding
  -- request or perform the preceding action
systemUnavail        (24),
  -- the request cannot be handled due to system unavailability
systemFailure        (25),
  -- the request cannot be handled due to system failure
duplicateCertReq     (26)
  -- certificate cannot be issued because a duplicate
  -- certificate already exists
}

PKIStatusInfo ::= SEQUENCE {
  status          PKIStatus,
  statusString    PKIFreeText    OPTIONAL,
  failInfo        PKIFailureInfo OPTIONAL
}

OOBCert ::= CMPCertificate

OOBCertHash ::= SEQUENCE {
  hashAlg    [0] AlgorithmIdentifier    OPTIONAL,
  certId     [1] CertId                  OPTIONAL,
  hashVal     BIT STRING
  -- hashVal is calculated over the DER encoding of the
  -- self-signed certificate with the identifier certID.
}

POPODecKeyChallContent ::= SEQUENCE OF Challenge
-- One Challenge per encryption key certification request (in the
-- same order as these requests appear in CertReqMessages).

Challenge ::= SEQUENCE {
  owf          AlgorithmIdentifier    OPTIONAL,
  -- MUST be present in the first Challenge; MAY be omitted in
  -- any subsequent Challenge in POPODecKeyChallContent (if
  -- omitted, then the owf used in the immediately preceding
  -- Challenge is to be used).
  witness      OCTET STRING,
  -- the result of applying the one-way function (owf) to a
```

```
-- randomly-generated INTEGER, A. [Note that a different
-- INTEGER MUST be used for each Challenge.]
challenge          OCTET STRING
-- the encryption (under the public key for which the cert.
-- request is being made) of Rand.
}

-- Added in CMP Updates [RFCXXXX]

Rand ::= SEQUENCE {
-- Rand is encrypted under the public key to form the challenge
-- in POPODecKeyChallContent
  int              INTEGER,
  -- the randomly-generated INTEGER A (above)
  sender           GeneralName
  -- the sender's name (as included in PKIHeader)
}

POPODecKeyRespContent ::= SEQUENCE OF INTEGER
-- One INTEGER per encryption key certification request (in the
-- same order as these requests appear in CertReqMessages). The
-- retrieved INTEGER A (above) is returned to the sender of the
-- corresponding Challenge.

CertRepMessage ::= SEQUENCE {
  caPubs           [1] SEQUENCE SIZE (1..MAX) OF CMPCertificate
                  OPTIONAL,
  response          SEQUENCE OF CertResponse
}

CertificationRequest ::= SEQUENCE {
  certificationRequestInfo SEQUENCE {
    version          INTEGER,
    subject          Name,
    subjectPublicKeyInfo SEQUENCE {
      algorithm      AlgorithmIdentifier,
      subjectPublicKey BIT STRING },
    attributes       [0] IMPLICIT SET OF Attribute },
  signatureAlgorithm AlgorithmIdentifier,
  signature           BIT STRING
}

CertResponse ::= SEQUENCE {
  certReqId        INTEGER,
  -- to match this response with corresponding request (a value
  -- of -1 is to be used if certReqId is not specified in the
  -- corresponding request, which can only be a pl0cr)
  status           PKIStatusInfo,
```

```
certifiedKeyPair    CertifiedKeyPair    OPTIONAL,
rspInfo            OCTET STRING        OPTIONAL
-- analogous to the id-regInfo-utf8Pairs string defined
-- for regInfo in CertReqMsg [RFC4211]
}

CertifiedKeyPair ::= SEQUENCE {
    certOrEncCert    CertOrEncCert,
    privateKey       [0] EncryptedKey    OPTIONAL,
    -- see [RFC4211] for comment on encoding
    -- Changed from Encrypted Value to EncryptedKey as a CHOICE of
    -- EncryptedValue and EnvelopedData due to the changes made in
    -- CMP Updates [RFCXXXX]
    -- Using the choice EncryptedValue is bit-compatible to the
    -- syntax without this change
    publicationInfo [1] PKIPublicationInfo OPTIONAL
}

CertOrEncCert ::= CHOICE {
    certificate       [0] CMPCertificate,
    encryptedCert     [1] EncryptedKey
    -- Changed from Encrypted Value to EncryptedKey as a CHOICE of
    -- EncryptedValue and EnvelopedData due to the changes made in
    -- CMP Updates [RFCXXXX]
    -- Using the choice EncryptedValue is bit-compatible to the
    -- syntax without this change
}

KeyRecRepContent ::= SEQUENCE {
    status            PKIStatusInfo,
    newSigCert        [0] CMPCertificate OPTIONAL,
    caCerts           [1] SEQUENCE SIZE (1..MAX) OF
                                CMPCertificate OPTIONAL,
    keyPairHist       [2] SEQUENCE SIZE (1..MAX) OF
                                CertifiedKeyPair OPTIONAL
}

RevReqContent ::= SEQUENCE OF RevDetails

RevDetails ::= SEQUENCE {
    certDetails       CertTemplate,
    -- allows requester to specify as much as they can about
    -- the cert. for which revocation is requested
    -- (e.g., for cases in which serialNumber is not available)
    crlEntryDetails   Extensions        OPTIONAL
    -- requested crlEntryExtensions
}
```

```
RevRepContent ::= SEQUENCE {
    status          SEQUENCE SIZE (1..MAX) OF PKIStatusInfo,
    -- in same order as was sent in RevReqContent
    revCerts [0] SEQUENCE SIZE (1..MAX) OF CertId
                                OPTIONAL,
    -- IDs for which revocation was requested
    -- (same order as status)
    crls           [1] SEQUENCE SIZE (1..MAX) OF CertificateList
                                OPTIONAL
    -- the resulting CRLs (there may be more than one)
}

CAKeyUpdAnnContent ::= SEQUENCE {
    oldWithNew      CMPCertificate, -- old pub signed with new priv
    newWithOld      CMPCertificate, -- new pub signed with old priv
    newWithNew      CMPCertificate -- new pub signed with new priv
}

CertAnnContent ::= CMPCertificate

RevAnnContent ::= SEQUENCE {
    status          PKIStatus,
    certId          CertId,
    willBeRevokedAt GeneralizedTime,
    badSinceDate    GeneralizedTime,
    crlDetails      Extensions OPTIONAL
    -- extra CRL details (e.g., crl number, reason, location, etc.)
}

CRLAnnContent ::= SEQUENCE OF CertificateList

CertConfirmContent ::= SEQUENCE OF CertStatus

CertStatus ::= SEQUENCE {
    certHash      OCTET STRING,
    -- the hash of the certificate, using the same hash algorithm
    -- as is used to create and verify the certificate signature
    certReqId     INTEGER,
    -- to match this confirmation with the corresponding req/rep
    statusInfo    PKIStatusInfo OPTIONAL,
    hashAlg [0] AlgorithmIdentifier OPTIONAL
    -- the hash algorithm to use for calculating certHash
    -- SHOULD NOT be used in all cases where the AlgorithmIdentifier
    -- of the certificate signature specifies a hash algorithm
}

PKIConfirmContent ::= NULL
```

```
-- CertReqTemplateContent, id-regCtrl-algId, id-regCtrl-algId, and
-- id-regCtrl-rsaKeyLen were added in CMP Updates [RFCXXXX]

CertReqTemplateContent ::= SEQUENCE {
    certTemplate          CertTemplate,
    -- prefilled certTemplate structure elements
    -- The SubjectPublicKeyInfo field in the certTemplate MUST NOT
    -- be used.
    keySpec               Controls OPTIONAL
    -- MAY be used to specify supported algorithms.
    -- Controls ::= SEQUENCE SIZE (1..MAX) OF AttributeTypeAndValue
    -- as specified in CRMF (RFC4211)
}

id-regCtrl-altCertTemplate OBJECT IDENTIFIER ::= { id-regCtrl 7 }
AltCertTemplate ::= AttributeTypeAndValue
-- specifies a template for a certificate other than an X.509v3
-- public-key certificate

id-regCtrl-algId OBJECT IDENTIFIER ::= { id-regCtrl 11 }
AlgIdCtrl ::= AlgorithmIdentifier
-- SHALL be used to specify supported algorithms other than RSA

id-regCtrl-rsaKeyLen OBJECT IDENTIFIER ::= { id-regCtrl 12 }
RsaKeyLenCtrl ::= INTEGER (1..MAX)
-- SHALL be used to specify supported RSA key lengths

-- RootCaKeyUpdateContent, CRLSource, and CRLStatus were added in
-- CMP Updates [RFCXXXX]

RootCaKeyUpdateContent ::= SEQUENCE {
    newWithNew            CMPCertificate,
    -- new root CA certificate
    newWithOld            [0] CMPCertificate OPTIONAL,
    -- X.509 certificate containing the new public root CA key
    -- signed with the old private root CA key
    oldWithNew            [1] CMPCertificate OPTIONAL
    -- X.509 certificate containing the old public root CA key
    -- signed with the new private root CA key
}

CRLSource ::= CHOICE {
    dpn                   [0] DistributionPointName,
    issuer                 [1] GeneralNames }

CRLStatus ::= SEQUENCE {
    source                 CRLSource,
    thisUpdate             Time OPTIONAL }
```



```

InfoTypeAndValue ::= SEQUENCE {
    infoType          OBJECT IDENTIFIER,
    infoValue         ANY DEFINED BY infoType OPTIONAL
}
-- Example InfoTypeAndValue contents include, but are not limited
-- to, the following (un-comment in this ASN.1 module and use as
-- appropriate for a given environment):
--
-- id-it-caProtEncCert      OBJECT IDENTIFIER ::= {id-it 1}
--   CAProtEncCertValue    ::= CMPCertificate
-- id-it-signKeyPairTypes  OBJECT IDENTIFIER ::= {id-it 2}
--   SignKeyPairTypesValue ::= SEQUENCE SIZE (1..MAX) OF
--                               AlgorithmIdentifier
-- id-it-encKeyPairTypes   OBJECT IDENTIFIER ::= {id-it 3}
--   EncKeyPairTypesValue  ::= SEQUENCE SIZE (1..MAX) OF
--                               AlgorithmIdentifier
-- id-it-preferredSymmAlg  OBJECT IDENTIFIER ::= {id-it 4}
--   PreferredSymmAlgValue ::= AlgorithmIdentifier
-- id-it-caKeyUpdateInfo   OBJECT IDENTIFIER ::= {id-it 5}
--   CAKeyUpdateInfoValue  ::= CAKeyUpdAnnContent
-- id-it-currentCRL        OBJECT IDENTIFIER ::= {id-it 6}
--   CurrentCRLValue       ::= CertificateList
-- id-it-unsupportedOIDs   OBJECT IDENTIFIER ::= {id-it 7}
--   UnsupportedOIDsValue  ::= SEQUENCE SIZE (1..MAX) OF
--                               OBJECT IDENTIFIER
-- id-it-keyPairParamReq   OBJECT IDENTIFIER ::= {id-it 10}
--   KeyPairParamReqValue  ::= OBJECT IDENTIFIER
-- id-it-keyPairParamRep   OBJECT IDENTIFIER ::= {id-it 11}
--   KeyPairParamRepValue  ::= AlgorithmIdentifier
-- id-it-revPassphrase     OBJECT IDENTIFIER ::= {id-it 12}
--   RevPassphraseValue    ::= EncryptedKey
--   - Changed from Encrypted Value to EncryptedKey as a CHOICE
--   - of EncryptedValue and EnvelopedData due to the changes
--   - made in CMP Updates [RFCXXXX]
--   - Using the choice EncryptedValue is bit-compatible to the
--   - syntax without this change
-- id-it-implicitConfirm  OBJECT IDENTIFIER ::= {id-it 13}
--   ImplicitConfirmValue  ::= NULL
-- id-it-confirmWaitTime  OBJECT IDENTIFIER ::= {id-it 14}
--   ConfirmWaitTimeValue  ::= GeneralizedTime
-- id-it-origPKIMessage   OBJECT IDENTIFIER ::= {id-it 15}
--   OrigPKIMessageValue   ::= PKIMessages
-- id-it-supplLangTags    OBJECT IDENTIFIER ::= {id-it 16}
--   SupplLangTagsValue    ::= SEQUENCE OF UTF8String
-- id-it-caCerts          OBJECT IDENTIFIER ::= {id-it 17}
--   CaCertsValue          ::= SEQUENCE SIZE (1..MAX) OF
--                               CMPCertificate
--   - id-it-caCerts added in CMP Updates [RFCXXXX]

```

```

-- id-it-rootCaKeyUpdate OBJECT IDENTIFIER ::= {id-it 18}
--   RootCaKeyUpdateValue ::= RootCaKeyUpdateContent
--   - id-it-rootCaKeyUpdate added in CMP Updates [RFCXXXX]
-- id-it-certReqTemplate OBJECT IDENTIFIER ::= {id-it 19}
--   CertReqTemplateValue ::= CertReqTemplateContent
--   - id-it-certReqTemplate added in CMP Updates [RFCXXXX]
-- id-it-rootCaCert OBJECT IDENTIFIER ::= {id-it 20}
--   RootCaCertValue ::= CMPCertificate
--   - id-it-rootCaCert added in CMP Updates [RFCXXXX]
-- id-it-certProfile OBJECT IDENTIFIER ::= {id-it 21}
--   CertProfileValue ::= SEQUENCE SIZE (1..MAX) OF
--                       UTF8String
--   - id-it-certProfile added in CMP Updates [RFCXXXX]
-- id-it-crlStatusList OBJECT IDENTIFIER ::= {id-it 22}
--   CRLStatusListValue ::= SEQUENCE SIZE (1..MAX) OF
--                           CRLStatus
--   - id-it-crlStatusList added in CMP Updates [RFCXXXX]
-- id-it-crls OBJECT IDENTIFIER ::= {id-it 23}
--   CRLsValue ::= SEQUENCE SIZE (1..MAX) OF
--                   CertificateList
--   - id-it-crls added in CMP Updates [RFCXXXX]
--
-- where
--
--   id-pkix OBJECT IDENTIFIER ::= {
--     iso(1) identified-organization(3)
--     dod(6) internet(1) security(5) mechanisms(5) pkix(7) }
-- and
--   id-it OBJECT IDENTIFIER ::= {id-pkix 4}
--
-- This construct MAY also be used to define new PKIX Certificate
-- Management Protocol request and response messages, or general-
-- purpose (e.g., announcement) messages for future needs or for
-- specific environments.

GenMsgContent ::= SEQUENCE OF InfoTypeAndValue

-- May be sent by EE, RA, or CA (depending on message content).
-- The OPTIONAL infoValue parameter of InfoTypeAndValue will
-- typically be omitted for some of the examples given above.
-- The receiver is free to ignore any contained OBJ. IDs that it
-- does not recognize. If sent from EE to CA, the empty set
-- indicates that the CA may send
-- any/all information that it wishes.

GenRepContent ::= SEQUENCE OF InfoTypeAndValue
-- Receiver MAY ignore any contained OIDs that it does not

```

```
-- recognize.

ErrorMsgContent ::= SEQUENCE {
    pKIStatusInfo      PKISStatusInfo,
    errorCode           INTEGER          OPTIONAL,
    -- implementation-specific error codes
    errorDetails        PKIFreeText      OPTIONAL
    -- implementation-specific error details
}

PollReqContent ::= SEQUENCE OF SEQUENCE {
    certReqId           INTEGER
}

PollRepContent ::= SEQUENCE OF SEQUENCE {
    certReqId           INTEGER,
    checkAfter          INTEGER, -- time in seconds
    reason              PKIFreeText OPTIONAL
}

--
-- Extended Key Usage extension for PKI entities used in CMP
-- operations, added due to the changes made in
-- CMP Updates [RFCXXXX]
-- The EKUs for the CA and RA are reused from CMC as defined in
-- [RFC6402]
--

-- id-kp-cmcCA OBJECT IDENTIFIER ::= { id-kp 27 }
-- id-kp-cmcRA OBJECT IDENTIFIER ::= { id-kp 28 }
id-kp-cmKGA OBJECT IDENTIFIER ::= { id-kp 32 }

-- There is no 1988 ASN.1 module of PKCS#9 available to import the
-- syntax of the localKeyId attribute type and value from. Therefore,
-- the syntax is added here as needed for the updates made in
-- CMP Updates [RFCXXXX]

pkcs-9 OBJECT IDENTIFIER ::= {iso(1) member-body(2) us(840)
                                rsadsi(113549) pkcs(1) 9}

pkcs-9-at-localKeyId OBJECT IDENTIFIER ::= {pkcs-9 21}

LocalKeyIdValue ::= OCTET STRING

END -- of CMP module
```

A.2. Update to RFC5912 - 2002 ASN.1 Module

This section contains the updated 2002 ASN.1 module for [RFC5912]. This module replaces the module in Section 9 of [RFC5912]. The module contains those changes to the normative ASN.1 module from RFC4210 Appendix F [RFC4210] that were to update to 2002 ASN.1 standard done in [RFC5912] as well as changes made in this document.

```
PKIXCMP-2021
    { iso(1) identified-organization(3) dod(6) internet(1)
      security(5) mechanisms(5) pkix(7) id-mod(0)
      id-mod-cmp2021-02(100) }
DEFINITIONS EXPLICIT TAGS ::=
BEGIN
IMPORTS

AttributeSet{}, SingleAttribute{}, Extensions{}, EXTENSION, ATTRIBUTE
FROM PKIX-CommonTypes-2009
    { iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-pkixCommon-02(57) }

AlgorithmIdentifier{}, SIGNATURE-ALGORITHM, ALGORITHM,
    DIGEST-ALGORITHM, MAC-ALGORITHM
FROM AlgorithmInformation-2009
    { iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0)
      id-mod-algorithmInformation-02(58) }

Certificate, CertificateList, Time, id-kp
FROM PKIX1Explicit-2009
    { iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-explicit-02(51) }

DistributionPointName, GeneralNames, GeneralName, KeyIdentifier
FROM PKIX1Implicit-2009
    { iso(1) identified-organization(3) dod(6) internet(1) security(5)
      mechanisms(5) pkix(7) id-mod(0) id-mod-pkix1-implicit-02(59) }

CertTemplate, PKIPublicationInfo, EncryptedKey, CertId,
    CertReqMessages, Controls, RegControlSet, id-regCtrl
FROM PKIXCRMF-2009
    { iso(1) identified-organization(3) dod(6) internet(1)
      security(5) mechanisms(5) pkix(7) id-mod(0)
      id-mod-crmf2005-02(55) }
-- The import of EncryptedKey is added due to the updates made
-- in CMP Updates [RFCXXXX]. EncryptedValue does not need to
-- be imported anymore and is therefore removed here.
```

```
-- see also the behavioral clarifications to CRMF codified in
-- Appendix C of this specification

CertificationRequest
FROM PKCS-10
    {iso(1) identified-organization(3) dod(6) internet(1) security(5)
    mechanisms(5) pkix(7) id-mod(0) id-mod-pkcs10-2009(69)}
-- (specified in RFC 2986 with 1993 ASN.1 syntax and IMPLICIT
-- tags). Alternatively, implementers may directly include
-- the [RFC2986] syntax in this module

localKeyId
FROM PKCS-9
    {iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
    modules(0) pkcs-9(1)}
-- The import of localKeyId is added due to the updates made in
-- CMP Updates [RFCXXXX]

EnvelopedData, SignedData
FROM CryptographicMessageSyntax-2009
    {iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
    smime(16) modules(0) id-mod-cms-2004-02(41)}
-- The import of EnvelopedData and SignedData is added due to
-- the updates made in CMP Updates [RFCXXXX]
;

-- the rest of the module contains locally defined OIDs and
-- constructs

CMPCertificate ::= CHOICE { x509v3PKCert Certificate, ... }
-- This syntax, while bits-on-the-wire compatible with the
-- standard X.509 definition of "Certificate", allows the
-- possibility of future certificate types (such as X.509
-- attribute certificates, WAP WTLS certificates, or other kinds
-- of certificates) within this certificate management protocol,
-- should a need ever arise to support such generality. Those
-- implementations that do not foresee a need to ever support
-- other certificate types MAY, if they wish, comment out the
-- above structure and "uncomment" the following one prior to
-- compiling this ASN.1 module. (Note that interoperability
-- with implementations that don't do this will be unaffected by
-- this change.)

-- CMPCertificate ::= Certificate

PKIMessage ::= SEQUENCE {
    header          PKIHeader,
    body            PKIBody,
```

```

protection    [0] PKIProtection OPTIONAL,
extraCerts    [1] SEQUENCE SIZE (1..MAX) OF CMPCertificate
               OPTIONAL }

```

```
PKIMessages ::= SEQUENCE SIZE (1..MAX) OF PKIMessage
```

```

PKIHeader ::= SEQUENCE {
    pvno                INTEGER          { cmp1999(1), cmp2000(2),
                                         cmp2012(3) },
    sender               GeneralName,
    -- identifies the sender
    recipient            GeneralName,
    -- identifies the intended recipient
    messageTime          [0] GeneralizedTime          OPTIONAL,
    -- time of production of this message (used when sender
    -- believes that the transport will be "suitable"; i.e.,
    -- that the time will still be meaningful upon receipt)
    protectionAlg        [1] AlgorithmIdentifier{ALGORITHM, {...}}
                           OPTIONAL,
    -- algorithm used for calculation of protection bits
    senderKID            [2] KeyIdentifier              OPTIONAL,
    recipKID            [3] KeyIdentifier              OPTIONAL,
    -- to identify specific keys used for protection
    transactionID        [4] OCTET STRING              OPTIONAL,
    -- identifies the transaction; i.e., this will be the same in
    -- corresponding request, response, certConf, and PKIConf
    -- messages
    senderNonce          [5] OCTET STRING              OPTIONAL,
    recipNonce          [6] OCTET STRING              OPTIONAL,
    -- nonces used to provide replay protection, senderNonce
    -- is inserted by the creator of this message; recipNonce
    -- is a nonce previously inserted in a related message by
    -- the intended recipient of this message
    freeText             [7] PKIFreeText              OPTIONAL,
    -- this may be used to indicate context-specific instructions
    -- (this field is intended for human consumption)
    generalInfo          [8] SEQUENCE SIZE (1..MAX) OF
                           InfoTypeAndValue          OPTIONAL
    -- this may be used to convey context-specific information
    -- (this field not primarily intended for human consumption)
}

```

```

PKIFreeText ::= SEQUENCE SIZE (1..MAX) OF UTF8String
    -- text encoded as UTF-8 String [RFC3629]

```

```

PKIBody ::= CHOICE {
    -- message-specific body elements
    ir    [0] CertReqMessages,      --Initialization Request
    ip    [1] CertRepMessage,       --Initialization Response

```

```

cr      [2]  CertReqMessages,      --Certification Request
cp      [3]  CertRepMessage,      --Certification Response
p10cr   [4]  CertificationRequest, --imported from [RFC2986]
popdecc [5]  POPODecKeyChallContent, --pop Challenge
popdecr [6]  POPODecKeyRespContent, --pop Response
kur      [7]  CertReqMessages,      --Key Update Request
kup      [8]  CertRepMessage,      --Key Update Response
krr      [9]  CertReqMessages,      --Key Recovery Request
krp     [10]  KeyRecRepContent,      --Key Recovery Response
rr       [11]  RevReqContent,        --Revocation Request
rp       [12]  RevRepContent,        --Revocation Response
ccr      [13]  CertReqMessages,      --Cross-Cert. Request
ccp      [14]  CertRepMessage,      --Cross-Cert. Response
ckuann   [15]  CAKeyUpdAnnContent,   --CA Key Update Ann.
cann     [16]  CertAnnContent,       --Certificate Ann.
rann     [17]  RevAnnContent,        --Revocation Ann.
crlann   [18]  CRLAnnContent,       --CRL Announcement
pkiconf  [19]  PKIConfirmContent,    --Confirmation
nested   [20]  NestedMessageContent, --Nested Message
genm     [21]  GenMsgContent,        --General Message
genp     [22]  GenRepContent,        --General Response
error    [23]  ErrorMsgContent,      --Error Message
certConf [24]  CertConfirmContent,   --Certificate confirm
pollReq  [25]  PollReqContent,       --Polling request
pollRep  [26]  PollRepContent        --Polling response
}

```

```

PKIProtection ::= BIT STRING

```

```

ProtectedPart ::= SEQUENCE {
    header    PKIHeader,
    body      PKIBody }

```

```

id-PasswordBasedMac OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    usa(840) nt(113533) nsn(7) algorithms(66) 13 }
PBMPParameter ::= SEQUENCE {
    salt          OCTET STRING,
    -- note: implementations MAY wish to limit acceptable sizes
    -- of this string to values appropriate for their environment
    -- in order to reduce the risk of denial-of-service attacks
    owf           AlgorithmIdentifier{DIGEST-ALGORITHM, {...}},
    -- AlgId for a One-Way Function
    iterationCount INTEGER,
    -- number of times the OWF is applied
    -- note: implementations MAY wish to limit acceptable sizes
    -- of this integer to values appropriate for their environment
    -- in order to reduce the risk of denial-of-service attacks
    mac           AlgorithmIdentifier{MAC-ALGORITHM, {...}}
}

```

```

-- the MAC AlgId (e.g., DES-MAC, Triple-DES-MAC [PKCS11],
-- or HMAC [RFC2104, RFC2202])
}

id-DHBasedMac OBJECT IDENTIFIER ::= { iso(1) member-body(2)
  usa(840) nt(113533) nsn(7) algorithms(66) 30 }
DHBMParameter ::= SEQUENCE {
  owf          AlgorithmIdentifier{DIGEST-ALGORITHM, {...}},
  -- AlgId for a One-Way Function
  mac          AlgorithmIdentifier{MAC-ALGORITHM, {...}}
  -- the MAC AlgId (e.g., DES-MAC, Triple-DES-MAC [PKCS11],
  -- or HMAC [RFC2104, RFC2202])
}

PKIStatus ::= INTEGER {
  accepted          (0),
  -- you got exactly what you asked for
  grantedWithMods   (1),
  -- you got something like what you asked for; the
  -- requester is responsible for ascertaining the differences
  rejection         (2),
  -- you don't get it, more information elsewhere in the message
  waiting           (3),
  -- the request body part has not yet been processed; expect to
  -- hear more later (note: proper handling of this status
  -- response MAY use the polling req/rep PKIMessages specified
  -- in Section 5.3.22 of [RFC4210]; alternatively, polling in the
  -- underlying transport layer MAY have some utility in this
  -- regard)
  revocationWarning (4),
  -- this message contains a warning that a revocation is
  -- imminent
  revocationNotification (5),
  -- notification that a revocation has occurred
  keyUpdateWarning  (6)
  -- update already done for the oldCertId specified in
  -- CertReqMsg
}

PKIFailureInfo ::= BIT STRING {
  -- since we can fail in more than one way!
  -- More codes may be added in the future if/when required.
  badAlg          (0),
  -- unrecognized or unsupported Algorithm Identifier
  badMessageCheck (1),
  -- integrity check failed (e.g., signature did not verify)
  badRequest      (2),
  -- transaction not permitted or supported

```



```
badTime          (3),
-- messageTime was not sufficiently close to the system time,
-- as defined by local policy
badCertId        (4),
-- no certificate could be found matching the provided criteria
badDataFormat    (5),
-- the data submitted has the wrong format
wrongAuthority    (6),
-- the authority indicated in the request is different from the
-- one creating the response token
incorrectData     (7),
-- the requester's data is incorrect (for notary services)
missingTimeStamp (8),
-- when the timestamp is missing but should be there
-- (by policy)
badPOP           (9),
-- the proof-of-possession failed
certRevoked      (10),
-- the certificate has already been revoked
certConfirmed    (11),
-- the certificate has already been confirmed
wrongIntegrity   (12),
-- not valid integrity, password based instead of signature or
-- vice versa
badRecipientNonce (13),
-- not valid recipient nonce, either missing or wrong value
timeNotAvailable (14),
-- the TSA's time source is not available
unacceptedPolicy (15),
-- the requested TSA policy is not supported by the TSA
unacceptedExtension (16),
-- the requested extension is not supported by the TSA
addInfoNotAvailable (17),
-- the additional information requested could not be
-- understood or is not available
badSenderNonce   (18),
-- not valid sender nonce, either missing or wrong size
badCertTemplate  (19),
-- not valid cert. template or missing mandatory information
signerNotTrusted (20),
-- signer of the message unknown or not trusted
transactionIdInUse (21),
-- the transaction identifier is already in use
unsupportedVersion (22),
-- the version of the message is not supported
notAuthorized    (23),
-- the sender was not authorized to make the preceding
-- request or perform the preceding action
```

```

    systemUnavail      (24),
    -- the request cannot be handled due to system unavailability
    systemFailure      (25),
    -- the request cannot be handled due to system failure
    duplicateCertReq    (26)
    -- certificate cannot be issued because a duplicate
    -- certificate already exists
}

PKIStatusInfo ::= SEQUENCE {
    status          PKIStatus,
    statusString     PKIFreeText      OPTIONAL,
    failInfo         PKIFailureInfo   OPTIONAL }

OOBCert ::= CMPCertificate

OOBCertHash ::= SEQUENCE {
    hashAlg          [0] AlgorithmIdentifier{DIGEST-ALGORITHM, {...}}
                        OPTIONAL,
    certId           [1] CertId          OPTIONAL,
    hashVal          BIT STRING
    -- hashVal is calculated over the DER encoding of the
    -- self-signed certificate with the identifier certID.
}

POPODecKeyChallContent ::= SEQUENCE OF Challenge
-- One Challenge per encryption key certification request (in the
-- same order as these requests appear in CertReqMessages).

Challenge ::= SEQUENCE {
    owf              AlgorithmIdentifier{DIGEST-ALGORITHM, {...}}
                        OPTIONAL,
    -- MUST be present in the first Challenge; MAY be omitted in
    -- any subsequent Challenge in POPODecKeyChallContent (if
    -- omitted, then the owf used in the immediately preceding
    -- Challenge is to be used).
    witness          OCTET STRING,
    -- the result of applying the one-way function (owf) to a
    -- randomly-generated INTEGER, A. [Note that a different
    -- INTEGER MUST be used for each Challenge.]
    challenge        OCTET STRING
    -- the encryption (under the public key for which the cert.
    -- request is being made) of Rand.
}

-- Added in CMP Updates [RFCXXXX]

Rand ::= SEQUENCE {

```

```

-- Rand is encrypted under the public key to form the challenge
-- in POPODecKeyChallContent
int                INTEGER,
-- the randomly-generated INTEGER A (above)
sender             GeneralName
-- the sender's name (as included in PKIHeader)
}

POPODecKeyRespContent ::= SEQUENCE OF INTEGER
-- One INTEGER per encryption key certification request (in the
-- same order as these requests appear in CertReqMessages). The
-- retrieved INTEGER A (above) is returned to the sender of the
-- corresponding Challenge.

CertRepMessage ::= SEQUENCE {
    caPubs          [1] SEQUENCE SIZE (1..MAX) OF CMPCertificate
                    OPTIONAL,
    response         SEQUENCE OF CertResponse }

CertResponse ::= SEQUENCE {
    certReqId        INTEGER,
    -- to match this response with the corresponding request (a value
    -- of -1 is to be used if certReqId is not specified in the
    -- corresponding request, which can only be a p10cr)
    status           PKIStatusInfo,
    certifiedKeyPair  CertifiedKeyPair    OPTIONAL,
    rspInfo          OCTET STRING         OPTIONAL
    -- analogous to the id-regInfo-utf8Pairs string defined
    -- for regInfo in CertReqMsg [RFC4211]
}

CertifiedKeyPair ::= SEQUENCE {
    certOrEncCert    CertOrEncCert,
    privateKey       [0] EncryptedKey    OPTIONAL,
    -- see [RFC4211] for comment on encoding
    -- Changed from Encrypted Value to EncryptedKey as a CHOICE of
    -- EncryptedValue and EnvelopedData due to the changes made in
    -- CMP Updates [RFCXXXX]
    -- Using the choice EncryptedValue is bit-compatible to the
    -- syntax without this change
    publicationInfo  [1] PKIPublicationInfo OPTIONAL }

CertOrEncCert ::= CHOICE {
    certificate       [0] CMPCertificate,
    encryptedCert     [1] EncryptedKey
    -- Changed from Encrypted Value to EncryptedKey as a CHOICE of
    -- EncryptedValue and EnvelopedData due to the changes made in
    -- CMP Updates [RFCXXXX]
}

```

```
-- Using the choice EncryptedValue is bit-compatible to the
-- syntax without this change
}

KeyRecRepContent ::= SEQUENCE {
    status                PKIStatusInfo,
    newSigCert             [0] CMPCertificate OPTIONAL,
    caCerts                [1] SEQUENCE SIZE (1..MAX) OF
                           CMPCertificate OPTIONAL,
    keyPairHist           [2] SEQUENCE SIZE (1..MAX) OF
                           CertifiedKeyPair OPTIONAL }

RevReqContent ::= SEQUENCE OF RevDetails

RevDetails ::= SEQUENCE {
    certDetails           CertTemplate,
    -- allows requester to specify as much as they can about
    -- the cert. for which revocation is requested
    -- (e.g., for cases in which serialNumber is not available)
    crlEntryDetails       Extensions{{...}} OPTIONAL
    -- requested crlEntryExtensions
}

RevRepContent ::= SEQUENCE {
    status                SEQUENCE SIZE (1..MAX) OF PKIStatusInfo,
    -- in same order as was sent in RevReqContent
    revCerts [0] SEQUENCE SIZE (1..MAX) OF CertId OPTIONAL,
    -- IDs for which revocation was requested
    -- (same order as status)
    crls                [1] SEQUENCE SIZE (1..MAX) OF CertificateList OPTIONAL
    -- the resulting CRLs (there may be more than one)
}

CAKeyUpdAnnContent ::= SEQUENCE {
    oldWithNew            CMPCertificate, -- old pub signed with new priv
    newWithOld            CMPCertificate, -- new pub signed with old priv
    newWithNew            CMPCertificate -- new pub signed with new priv
}

CertAnnContent ::= CMPCertificate

RevAnnContent ::= SEQUENCE {
    status                PKIStatus,
    certId                CertId,
    willBeRevokedAt       GeneralizedTime,
    badSinceDate          GeneralizedTime,
    crlDetails            Extensions{{...}} OPTIONAL
    -- extra CRL details (e.g., crl number, reason, location, etc.)
}
```

```
}

CRLAnnContent ::= SEQUENCE OF CertificateList
PKIConfirmContent ::= NULL

NestedMessageContent ::= PKIMessages

-- CertReqTemplateContent, AttributeTypeAndValue,
-- ExpandedRegControlSet, id-regCtrl-altCertTemplate,
-- AltCertTemplate, regCtrl-algId, id-regCtrl-algId, AlgIdCtrl,
-- regCtrl-rsaKeyLen, id-regCtrl-rsaKeyLen, and RsaKeyLenCtrl
-- were added in CMP Updates [RFCXXXX]

CertReqTemplateContent ::= SEQUENCE {
    certTemplate          CertTemplate,
    -- prefilled certTemplate structure elements
    -- The SubjectPublicKeyInfo field in the certTemplate MUST NOT
    -- be used.
    keySpec              Controls OPTIONAL
    -- MAY be used to specify supported algorithms.
    -- Controls ::= SEQUENCE SIZE (1..MAX) OF AttributeTypeAndValue
    -- as specified in CRMF (RFC4211)
}

AttributeTypeAndValue ::= SingleAttribute({ ... })

ExpandedRegControlSet ATTRIBUTE ::= { RegControlSet |
    regCtrl-altCertTemplate | regCtrl-algId | regCtrl-rsaKeyLen, ... }

regCtrl-altCertTemplate ATTRIBUTE ::=
    { TYPE AltCertTemplate IDENTIFIED BY id-regCtrl-altCertTemplate }

id-regCtrl-altCertTemplate OBJECT IDENTIFIER ::= { id-regCtrl 7 }

AltCertTemplate ::= AttributeTypeAndValue
    -- specifies a template for a certificate other than an X.509v3
    -- public-key certificate

regCtrl-algId ATTRIBUTE ::=
    { TYPE AlgIdCtrl IDENTIFIED BY id-regCtrl-algId }

id-regCtrl-algId OBJECT IDENTIFIER ::= { id-regCtrl 11 }

AlgIdCtrl ::= AlgorithmIdentifier(ALGORITHM, {...})
    -- SHALL be used to specify supported algorithms other than RSA

regCtrl-rsaKeyLen ATTRIBUTE ::=
    { TYPE RsaKeyLenCtrl IDENTIFIED BY id-regCtrl-rsaKeyLen }
```

```

id-regCtrl-rsaKeyLen OBJECT IDENTIFIER ::= { id-regCtrl 12 }

RsaKeyLenCtrl ::= INTEGER (1..MAX)
    -- SHALL be used to specify supported RSA key lengths

-- RootCaKeyUpdateContent, CRLSource, and CRLStatus were added in
-- CMP Updates [RFCXXXX]

RootCaKeyUpdateContent ::= SEQUENCE {
    newWithNew      CMPCertificate,
    -- new root CA certificate
    newWithOld      [0] CMPCertificate OPTIONAL,
    -- X.509 certificate containing the new public root CA key
    -- signed with the old private root CA key
    oldWithNew      [1] CMPCertificate OPTIONAL,
    -- X.509 certificate containing the old public root CA key
    -- signed with the new private root CA key
}

CRLSource ::= CHOICE {
    dpn             [0] DistributionPointName,
    issuer           [1] GeneralNames }

CRLStatus ::= SEQUENCE {
    source           CRLSource,
    thisUpdate       Time OPTIONAL }

INFO-TYPE-AND-VALUE ::= TYPE-IDENTIFIER

InfoTypeAndValue ::= SEQUENCE {
    infoType         INFO-TYPE-AND-VALUE.
                      &id({SupportedInfoSet}),
    infoValue        INFO-TYPE-AND-VALUE.
                      &Type({SupportedInfoSet}{@infoType}) }

SupportedInfoSet INFO-TYPE-AND-VALUE ::= { ... }

-- Example InfoTypeAndValue contents include, but are not limited
-- to, the following (uncomment in this ASN.1 module and use as
-- appropriate for a given environment):
--
-- id-it-caProtEncCert      OBJECT IDENTIFIER ::= {id-it 1}
--   CAProtEncCertValue     ::= CMPCertificate
-- id-it-signKeyPairTypes  OBJECT IDENTIFIER ::= {id-it 2}
--   SignKeyPairTypesValue  ::= SEQUENCE SIZE (1..MAX) OF
--                               AlgorithmIdentifier{...}
-- id-it-encKeyPairTypes   OBJECT IDENTIFIER ::= {id-it 3}
--   EncKeyPairTypesValue   ::= SEQUENCE SIZE (1..MAX) OF

```

```

--                                     AlgorithmIdentifier({...})
-- id-it-preferredSymmAlg OBJECT IDENTIFIER ::= {id-it 4}
--   PreferredSymmAlgValue ::= AlgorithmIdentifier({...})
-- id-it-caKeyUpdateInfo OBJECT IDENTIFIER ::= {id-it 5}
--   CAKeyUpdateInfoValue ::= CAKeyUpdAnnContent
-- id-it-currentCRL OBJECT IDENTIFIER ::= {id-it 6}
--   CurrentCRLValue ::= CertificateList
-- id-it-unsupportedOIDs OBJECT IDENTIFIER ::= {id-it 7}
--   UnsupportedOIDsValue ::= SEQUENCE SIZE (1..MAX) OF
--                                     OBJECT IDENTIFIER
-- id-it-keyPairParamReq OBJECT IDENTIFIER ::= {id-it 10}
--   KeyPairParamReqValue ::= OBJECT IDENTIFIER
-- id-it-keyPairParamRep OBJECT IDENTIFIER ::= {id-it 11}
--   KeyPairParamRepValue ::= AlgorithmIdentifier({...})
-- id-it-revPassphrase OBJECT IDENTIFIER ::= {id-it 12}
--   RevPassphraseValue ::= EncryptedKey
--   - Changed from Encrypted Value to EncryptedKey as a CHOICE
--   - of EncryptedValue and EnvelopedData due to the changes
--   - made in CMP Updates [RFCXXXX]
--   - Using the choice EncryptedValue is bit-compatible to
--   - the syntax without this change
-- id-it-implicitConfirm OBJECT IDENTIFIER ::= {id-it 13}
--   ImplicitConfirmValue ::= NULL
-- id-it-confirmWaitTime OBJECT IDENTIFIER ::= {id-it 14}
--   ConfirmWaitTimeValue ::= GeneralizedTime
-- id-it-origPKIMessage OBJECT IDENTIFIER ::= {id-it 15}
--   OrigPKIMessageValue ::= PKIMessages
-- id-it-supplLangTags OBJECT IDENTIFIER ::= {id-it 16}
--   SupplLangTagsValue ::= SEQUENCE OF UTF8String
-- id-it-caCerts OBJECT IDENTIFIER ::= {id-it 17}
--   CaCertsValue ::= SEQUENCE SIZE (1..MAX) OF
--                                     CMPCertificate
--   - id-it-caCerts added in CMP Updates [RFCXXXX]
-- id-it-rootCaKeyUpdate OBJECT IDENTIFIER ::= {id-it 18}
--   RootCaKeyUpdateValue ::= RootCaKeyUpdateContent
--   - id-it-rootCaKeyUpdate added in CMP Updates [RFCXXXX]
-- id-it-certReqTemplate OBJECT IDENTIFIER ::= {id-it 19}
--   CertReqTemplateValue ::= CertReqTemplateContent
--   - id-it-certReqTemplate added in CMP Updates [RFCXXXX]
-- id-it-rootCaCert OBJECT IDENTIFIER ::= {id-it 20}
--   RootCaCertValue ::= CMPCertificate
--   - id-it-rootCaCert added in CMP Updates [RFCXXXX]
-- id-it-certProfile OBJECT IDENTIFIER ::= {id-it 21}
--   CertProfileValue ::= SEQUENCE SIZE (1..MAX) OF
--                                     UTF8String
--   - id-it-certProfile added in CMP Updates [RFCXXXX]
-- id-it-crlStatusList OBJECT IDENTIFIER ::= {id-it 22}
--   CRLStatusListValue ::= SEQUENCE SIZE (1..MAX) OF

```

```
--                                     CRLStatus
--      - id-it-crlStatusList added in CMP Updates [RFCXXXX]
--      id-it-crls      OBJECT IDENTIFIER ::= {id-it 23}
--      CRLsValue      ::= SEQUENCE SIZE (1..MAX) OF
--                          CertificateList
--      - id-it-crls added in CMP Updates [RFCXXXX]
--
-- where
--
--      id-pkix OBJECT IDENTIFIER ::= {
--          iso(1) identified-organization(3)
--          dod(6) internet(1) security(5) mechanisms(5) pkix(7)}
-- and
--      id-it      OBJECT IDENTIFIER ::= {id-pkix 4}
--
-- This construct MAY also be used to define new PKIX Certificate
-- Management Protocol request and response messages, or general-
-- purpose (e.g., announcement) messages for future needs or for
-- specific environments.
```

GenMsgContent ::= SEQUENCE OF InfoTypeAndValue

```
-- May be sent by EE, RA, or CA (depending on message content).
-- The OPTIONAL infoValue parameter of InfoTypeAndValue will
-- typically be omitted for some of the examples given above.
-- The receiver is free to ignore any contained OBJECT IDs that it
-- does not recognize.  If sent from EE to CA, the empty set
-- indicates that the CA may send
-- any/all information that it wishes.
```

GenRepContent ::= SEQUENCE OF InfoTypeAndValue

```
-- Receiver MAY ignore any contained OIDs that it does not
-- recognize.
```

```
ErrorMsgContent ::= SEQUENCE {
    pKIStatusInfo      PKISStatusInfo,
    errorCode           INTEGER          OPTIONAL,
    -- implementation-specific error codes
    errorDetails        PKIFreeText      OPTIONAL
    -- implementation-specific error details
}
```

CertConfirmContent ::= SEQUENCE OF CertStatus

```
CertStatus ::= SEQUENCE {
    certHash      OCTET STRING,
    -- the hash of the certificate, using the same hash algorithm
```



```
-- as is used to create and verify the certificate signature
certReqId  INTEGER,
-- to match this confirmation with the corresponding req/rep
statusInfo PKIStatusInfo OPTIONAL,
hashAlg [0] AlgorithmIdentifier{DIGEST-ALGORITHM, {...}} OPTIONAL
-- the hash algorithm to use for calculating certHash
-- SHOULD NOT be used in all cases where the AlgorithmIdentifier
-- of the certificate signature specifies a hash algorithm
}

PollReqContent ::= SEQUENCE OF SEQUENCE {
    certReqId      INTEGER }

PollRepContent ::= SEQUENCE OF SEQUENCE {
    certReqId      INTEGER,
    checkAfter     INTEGER, -- time in seconds
    reason         PKIFreeText OPTIONAL }

--
-- Extended Key Usage extension for PKI entities used in CMP
-- operations, added due to the changes made in
-- CMP Updates [RFCXXXX]
-- The EKUs for the CA and RA are reused from CMC as defined in
-- [RFC6402]
--

-- id-kp-cmcCA OBJECT IDENTIFIER ::= { id-kp 27 }
-- id-kp-cmcRA OBJECT IDENTIFIER ::= { id-kp 28 }
id-kp-cmKGA OBJECT IDENTIFIER ::= { id-kp 32 }

END
```

Appendix B. History of Changes

[RFC Editor: This appendix must be deleted in the final version of the document.]

From version 22 -> 23:

- * Addressed comments from IESG discussion (see thread "Francesca Palombini's No Objection on draft-ietf-lamps-cmp-updates-22: (with COMMENT)")
- * Addressed comment from Carl (see thread "Paul Wouters' Discuss on draft-ietf-lamps-cmp-updates-21: (with DISCUSS and COMMENT)")

From version 21 -> 22:

- * Addressed comments from IESG discussion (see thread " Paul Wouters' Discuss on draft-ietf-lamps-cmp-updates-21: (with DISCUSS and COMMENT) ")

From version 20 -> 21:

- * Extended Section 1 based on feedback from the IESG telechat
- * Removed a redundant paragraph from the Abstract

From version 19 -> 20:

- * Addressed comments reported after GEN AD review

From version 18 -> 19:

- * Deleted the Comments on IANA ToDos and changed the decimals TBD1 -> 22 and TBD2 -> 23
- * Updated Section 3.4 regarding ToDos updating the well-known URI registration

From version 17 -> 18:

- * Addressed comments from AD Evaluation (see thread "AD Review of draft-ietf-lamps-cmp-updates-17")
- * Added Section 2.8 to clarify on the usage of GeneralizedTime (see thread "draft-ietf-lamps-cmp-updates: fractional seconds")
- * Updated Section 3.4 introducing the path segment 'p' to indicate the following arbitrary label according to the discussion during IETF 113 (see thread "/.well-known/brski reference to brski-registry")
- * Capitalized all headlines

From version 16 -> 17:

- * Removed the pre-RFC5378 work disclaimer after the RFC 4210 authors granted BCP78 rights to the IETF Trust
- * Removed note on usage of language tags in UTF8String due to reference to references to outdated/historic RFCs
- * Resolved some nits reported by I-D nit checker tool

From version 15 -> 16:

- * Updated IPR disclaimer

From version 14 -> 15:

- * Updated Section 2.16 clarifying the usage of CRLSource (see thread "CRL update retrieval - WG Last Call for draft-ietf-lamps-cmp-updates-14 and draft-ietf-lamps-lightweight-cmp-profile-08")
- * Updated Section 2.22 adding further references regarding random number generation (see thread "CMP draft WGLC: measuring entropy, CA certificates")
- * Fixed some nits

From version 13 -> 14:

- * Extended id-it-caCerts support message to allow transporting to-be-trusted root CA certificates; added respective security consideration (see thread "Generalizing the CMP "Get CA certificates" use case")
- * Rolled back changes made in previous version regarding root CA update to avoid registration of new OIDs. Yet we stucked to using id-it-rootCaCert in the genm body instead its headers' generalInfo field and removed the Todos and TBDs on re-arranging id-it OIDs (see thread "Allocation of OIDs for CRL update retrieval (draft-ietf-lamps-cmp-updates-13)")

From version 12 -> 13:

- * Added John Gray to the list of authors due to fruitful discussion and important proposals
- * Fixed errata no. 2615, 2616, 3949, 4078, and 5201 on RFC 4210
- * Added reference on RFC 8933 regarding CMS signedAttrs to Section 2.7
- * Updated Section 2.9 and the ASN.1 modules moving the position of the hashAlg field (see thread "[CMP Updates] position of hashAlg in certStatus")
- * Changed "rootCaCert" from generalInfo to genm body and generalized to "oldTrustAnchor", renaming "rootCaKeyUpdate" to "trustAnchorUpdate" in Sections 2.14, A.1, and A.2, removing former Section 2.4
- * Added genm use case "CRL update retrieval" in Section 2.16, A.1, and A.2. (see thread "[CMP Updates] Requesting a current CRL")
- * Updated Section 2.18 and 2.17 to support polling for all kinds of CMP request messages initiated by an error message with status "waiting" as initially discussed at IETF 111
- * Updated Sections 2.19 and 2.20 regarding version handling
- * Added further OIDs and a TBD regarding reordering of the OIDs
- * Added Sections 2.21 to 2.23 with new security considerations and updated Section 5 accordingly
- * Added a ToDo regarding OID registration, renaming, and re-ordering
- * Added Section 3.1 updating the introduction of RFC 6712

- * Fixed some nits in the ASN.1 modules (see thread "draft-ietf-lamps-cmp-updates-12: Comments on A.1. 1988 ASN.1 Module" and "draft-ietf-lamps-cmp-updates-12: Comments on A.2. 2002 ASN.1 Module")
- * Replaced the term "transport" by "transfer" where appropriate to prevent confusion
- * Minor editorial changes

From version 11 -> 12:

- * Extended Section 2.5 and the ASN.1 modules in Appendix A to allow a sequence of certificate profiles in CertProfileValue (see thread "id-it-CertProfile in draft-ietf-lamps-cmp-updates")

From version 10 -> 11:

- * Add Section 2.10 to add an additional hashAlg field to the CertStatus type to support certificates signed with a signature algorithm not explicitly indicating a hash algorithm in the AlgorithmIdentifier (see thread "Hash algorithm to us for calculating certHash")
- * Added newly registered OIDs and temporarily registered URI suffix
- * Exchanged the import of CertificationRequest from RFC 2986 to the definition from RFC 6402 Appendix A.1 (see thread "CMP Update of CertificationRequest")
- * Corrected the definition of LocalKeyIdValue in Appendix A.1
- * Updated new RFC numbers for draft-lamps-crmf-update-algs

From version 9 -> 10:

- * Added 1988 ASN.1 syntax for localKeyId attribute to Appendix A.1

From version 08 -> 09:

- * Deleted specific definition of CMP CA and CMP RA in Section 2.2 and only reference RFC 6402 for definition of id-kp-cmCCA and id-kp-cmcRA to resolve the ToDo below based on feedback of Tomas Gustavsson
- * Added Section 2.4. and 2.5 to define id-it-rootCaCert and id-it-certProfile to be used in Section 2.14 and 2.15
- * Added reference to CMP Algorithms in Section 2.8
- * Extended Section 2.14 to explicitly indicate the root CA an update is requested for by using id-it-rootCaCert and changing the ASN.1 syntax to require providing the newWithOld certificate in the response message
- * Extended Section 2.15 to explicitly indicate the certificate request template by using id-it-certProfile and on further details of the newly introduced controls

- * Deleted the table on id-kp-cmcCA and id-kp-cmcRA and adding id-it-rootCaCert and id-it-certProfile in Section 2.19
- * Adding the definition of id-it-rootCaCert and id-it-certProfile in both ASN.1 modules in Appendix A
- * Minor editorial changes reflecting the above changes

From version 07 -> 08:

- * Added a ToDo to Section 2.2 to reflect a current discussion on the need of an additional CMP-CA role and ECU and differentiation from CMP-RA
- * Added ToDos to Section 2.12 and 2.13

From version 06 -> 07:

- * Added David von Oheimb as co-author
- * Changed to XML V3
- * Added Section 2.3 to enable a CMP protocol version number 3 in the PKIHeader for cases where EnvelopedData is to be used (see thread "Mail regarding draft-ietf-lamps-cmp-updates").
- * Added Section 2.4 to refer to draft-ietf-lamps-crmf-update-algs for the update of id-PasswordBasedMac for PKI message protection using passwords or shared secrets.
- * Updated Section 2.6 to introduce the protocol version number 3 to properly indicate support of EnvelopedData instead of EncryptedValue in case a transaction requires use of EnvelopedData (see thread "Mail regarding draft-ietf-lamps-cmp-updates").
- * Update Section 2.14 to make the minimal changes to the respective section in CMP more explicit.
- * Added Sections 2.15 and 2.16 to address the new cmp2021 protocol version in Section 7 Version Negotiation.
- * Updated Section 2.17 to add new OIDs for id-regCtrl-algId and id-regCtrl-rsaKeyLen for registration at IANA.
- * Added Section 2.20 to update the general rules of interpretation in Appendix D.1 regarding the new cmp2021 version.
- * Added Section 2.21 to update the Algorithm Use Profile in Appendix D.2 with the reference to the new CMP Algorithms document as decided at IETF 108.
- * Updates Section 3.1 to delete the description of a discovery mechanism as decided at IETF 108.
- * Various changes and corrections in wording.

From version 05 -> 06:

- * Added the update of Appendix D.2 with the reference to the new CMP Algorithms document as decided in IETF 108
- * Updated the IANA considerations to register new OIDs for id-regCtrl-algId and d-regCtrl-rsaKeyLen.

- * Minor changes and corrections

From version 04 -> 05:

- * Added Section 2.11 and Section 2.12 to clarify the usage of these general messages types with EC curves (see thread "AlgorithmIdentifier parameters NULL value - Re: InfoTypeAndValue in CMP headers")
- * Split former section 2.7 on adding 'CA Certificates', 'Root CA Certificates Update', and 'Certificate Request Template' in three separate sections for easier readability
- * Changed in Section 2.15 the ASN.1 syntax of CertReqTemplateValue from using rsaKeyLen to usage of controls as specified in CRMF Section 6 [RFC4211] (see thread "draft-ietf-lamps-cmp-updates and rsaKeyLen")
- * Updated the IANA considerations in Section 4 to introduce new OID for id-regCtrl-algId and id-regCtrl-rsaKeyLen (see thread "draft-ietf-lamps-cmp-updates and rsaKeyLen")
- * Updated the IANA Considerations in and the Appendixes to introduce new OID for the updates ASN.1 modules (see thread "I-D Action: draft-ietf-lamps-cmp-updates-04.txt")
- * Removed EncryptedValue from and added Controls to the list of types imported from CRMF [RFC4211] in ASN.1 modules (see thread "draft-ietf-lamps-cmp-updates and the ASN.1 modules")
- * Moved declaration of Rand out of the comment in ASN.1 modules (see thread "draft-ietf-lamps-cmp-updates and the ASN.1 modules")
- * Minor changes and corrections

From version 03 -> 04:

- * Added Section 2.7 to introduce three new id-it IDs for uses in general messages as discussed (see thread "draft-ietf-lamps-cmp-updates add section to introduce id-it-caCerts, id-it-rootCaKeyUpdate, and id-it-certReqTemplate")
- * Added the new id-it IDs and the /.well-known/cmp to the IANA Considerations of [RFC4210] in Section 2.9
- * Updated the IANA Considerations of [RFC4210] in Section 2.26
- * Some changes in wording on Section 3 due to review comments from Martin Peylo

From version 02 -> 03:

- * Added a ToDo on aligning with the CMP Algorithms draft that will be set up as decided in IETF 108
- * Updated section on Encrypted Values in Section 2.7 to add the AsymmetricKey Package structure to transport a newly generated private key as decided in IETF 108
- * Updated the IANA Considerations of [RFC4210] in Section 2.26

- * Added the pre-registered OID in Section 2.26 and the ASN.1 module
- * Added Section 3 to document the changes to RFC 6712 [RFC6712] regarding URI discovery and using the path-prefix of '/.well-known/' as discussed in IETF 108
- * Updated the IANA Considerations section
- * Added a complete updated ASN.1 module in 1988 syntax to update Appendix F of [RFC4210] and a complete updated ASN.1 module in 2002 syntax to update Section 9 of [RFC5912]
- * Minor changes in wording

From version 01 -> 02:

- * Updated section on ECU OIDs in Section 2.2 as decided in IETF 107
- * Changed from symmetric key-encryption to password-based key management technique in Section 2.7 as discussed with Russ and Jim on the mailing list
- * Defined the attribute containing the key identifier for the revocation passphrase in Section 2.26
- * Moved the change history to the Appendix

From version 00 -> 01:

- * Minor changes in wording

From draft-brockhaus-lamps-cmp-updates-03 -> draft-ietf-lamps-cmp-updates-00:

- * Changes required to reflect WG adoption

From version 02 -> 03:

- * Added some clarification in Section 2.1

From version 01 -> 02:

- * Added clarification to section on multiple protection
- * Added clarification on new EKUs after some exchange with Tomas Gustavsson
- * Reused OIDs from RFC 6402 [RFC6402] as suggested by Sean Turner at IETF 106
- * Added clarification on the field containing the key identifier for a revocation passphrase
- * Minor changes in wording

From version 00 -> 01:

- * Added a section describing the new extended key usages

- * Completed the section on changes to the specification of encrypted values
- * Added a section on clarification to Appendix D.4
- * Minor generalization in RFC 4210 [RFC4210] Sections 5.1.3.4 and 5.3.22
- * Minor changes in wording

Authors' Addresses

Hendrik Brockhaus (editor)
Siemens
Werner-von-Siemens-Strasse 1
80333 Munich
Germany
Email: hendrik.brockhaus@siemens.com
URI: <https://www.siemens.com>

David von Oheimb
Siemens
Werner-von-Siemens-Strasse 1
80333 Munich
Germany
Email: david.von.oheimb@siemens.com
URI: <https://www.siemens.com>

John Gray
Entrust
1187 Park Place
Minneapolis, MN 55379
United States of America
Email: john.gray@entrust.com
URI: <https://www.entrust.com>